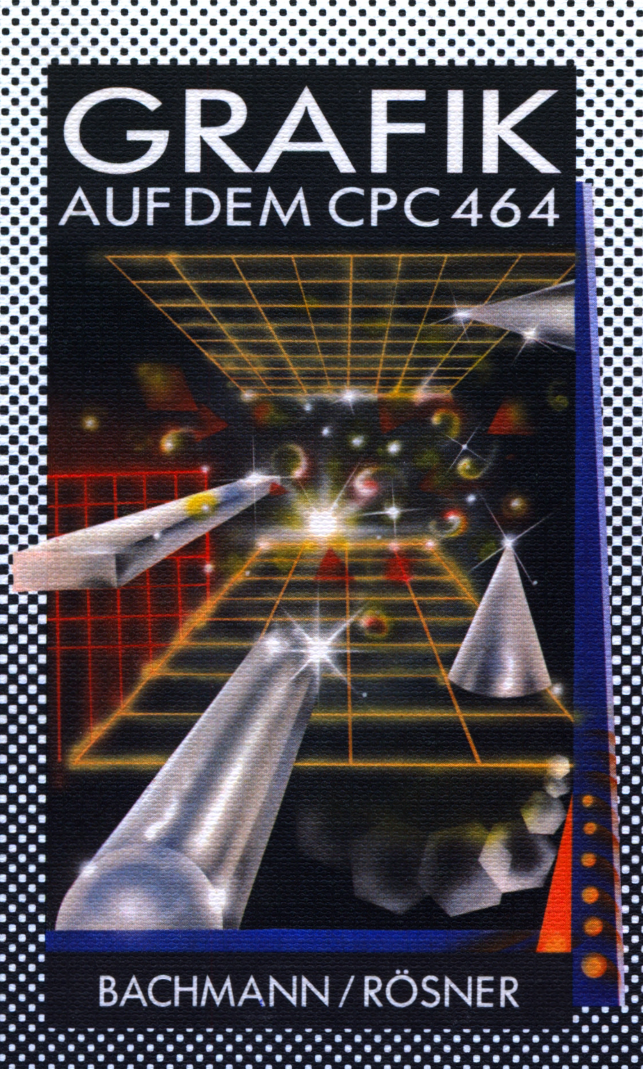
westermann COMPUTERBUCH

GRAFIK AUF DEM CPC 464



BACHMANN / RÖSNER

Walter Bachmann
Norbert Rösner

Grafik auf dem CPC 464

Eine Programmdiskette
ist unter der
Best.-Nr. **138018** erhältlich.

1988 1987 1986 1985
rstellung

aunschweig

lkmann
nschweig

Inhalt

Vorwort

Der Begriff Computergrafik kennzeichnet visuelle Darstellungen, die oft an künstlerische Ambitionen, aber auch an zweckbezogene, technische Grafiken erinnern. Computergrafik ist ein Teil der Informationsverarbeitung, bei der die Computerausgaben grafisch dargestellt werden. Gemäß dem Sprichwort „Ein Bild sagt mehr als 1000 Worte“ liefern technische Grafiken einen schnellen und umfassenden Überblick über komplizierte, funktionale Zusammenhänge. Der rasche Technologiefortschritt, verbunden mit der Massenproduktion von Chips führte dazu, daß bereits heute kostengünstige Computer grafikfähig sind.

Die Leistungsfähigkeit von grafikverarbeitenden Geräten (Rasterbildschirm, Plotter, Drukker, Lichtgriffel, Steuerknüppel, Rollkugel, grafische Tabletts u. a.) erfordert ein Lehrbuch, das die Grundlagen der Computergrafik behandelt. Das vorliegende Buch will in diese Grundlagen einführen. Die notwendigen geometrischen Voraussetzungen werden mit einfachen und anschaulichen Anleitungen aufbereitet. Die abgeleiteten Algorithmen werden in die Programmiersprache BASIC übertragen. Zu jedem Kapitel werden BASIC-Programme angeboten. Besonderer Wert wurde bei der Auswahl der Algorithmen auf eine Praxisnähe zu technisch-naturwissenschaftlichen Berufsfeldern gelegt. Obwohl alle Programme für den Schneider CPC 464 Computer geschrieben sind, ist eine leichte Übertragung der Programme auf andere Rechner möglich. Dies resultiert daher, daß der algorithmische Gedanke in diesem Buch gegenüber einer Ausnutzung von speziellen und selten gebrauchten Grafikbefehlen dominiert.

1. Bildschirmtechnologie

Es gibt verschiedene Arten von grafikfähigen Bildschirmen, die sich durch die Erzeugun;
des Bildes unterscheiden.

Ein **Vektorbildschirm** lenkt den Elektronenstrahl direkt ab, so daß durchgehende (nicht un
terbrochene) Linien auf dem Bildschirm erzeugt werden. Das Vektorverfahren gewährlei
stet eine gute Qualität bei hoher Auflösung. Bei einer Speicherbildröhre (Storage-tube
speichert eine Leuchtschicht (fluoreszierende Schicht) die geschriebenen Vektoren. Bei ei
nem Vektorbildschirm lassen sich die einzelnen Bildteile nicht herauslöschen. Bei der Bild
wiederholröhre (Refresh-Technik) werden die Vektoren 50mal pro Sekunde dargestellt
Mit zunehmender Bildfüllung können nicht alle Linien innerhalb eines Refresh-Zyklus ne
gezeichnet werden. Dadurch tritt ein Flimmern des Bildschirmes auf.

Bei einem **Rasterbildschirm** erfolgt eine zeilenweise Ablenkung des Elektronenstrahles
wobei innerhalb der Zeile eine abwechselnde Hell-Dunkel-Steuerung erfolgt. Dadurch er
geben sich Bildpunkte (Pixels). Die Auflösung des Rasterbildschirmes wird durch die An
zahl der schachbrettartig angeordneten Pixels bestimmt. Die hochauflösende Grafik de
CPC-464 weist 640 x 200 = 128 000 Bildpunkte auf. Jedem Bildpunkt ist im Rechner ei
Bit zugeordnet. Dem Bild entspricht ein bestimmtes Bitmuster (Bitmapping) im RAM
(Random Access Memory = Schreib-Lese-Speicher).

Ein Byte entspricht 8 Bit, d. h. 128 000 Bit entsprechen 16 000 Bytes. Diese knapp 16 KBy
te (1 KByte = 1024 Byte) für den hochauflösenden Bildschirm belegen den RAM-Bereic
von $C000 bis $FFFF (dezimal: 49 152 bis 65 535).

Im Gegensatz zum Vektorbildschirm kann der Rasterbildschirm, durch Setzen der entspre
chenden Bits, beinahe verzögerungsfrei aktualisiert werden. Bei Rastergrafik-Sichtgeräte
können Flächen äußerst schnell mit Grafikdaten aufgefüllt werden. Das Rasterverfahren er
möglicht gegenüber dem Vektorverfahren die Darstellung einer größeren Anzahl von Far
ben. Natürlich wächst mit der Farbenvielfalt der zugehörige Speicherbedarf. Der Raster
bildschirm ist vom Verfahren her gesehen flimmerfrei.

2. Zeichnen von Linien

In dem folgenden Testprogramm werden lediglich einige Linien auf dem Bildschirm dargestellt. Diese Linien sind durch den Anfangspunkt P1(x1,y1) und den Endpunkt P2(x2,y2) festgelegt. Die linke untere Ecke des Bildschirmes bildet den Nullpunkt des „Bildschirm-Koordinatensystems“, das in x-Richtung (von links nach rechts) die Punkte 0,1,2,. . .,639 enthält und in y-Richtung von 0 (linke untere Ecke) bis 399 (linke obere Ecke) läuft.

Zum Zeichnen einer Linie können die Anweisungen

```
MOVE x1,y1 : DRAW x2,y2, 1
```

verwendet werden. Zunächst wird der Grafik-Cursor gemäß den Koordinaten x1, y1 positioniert. Dann wird von dort aus eine Linie zum Punkt P2 mit den Koordinaten x2 und y2 gezeichnet. Eine Zahl 1 (bzw. 0) am Ende dieser DRAW-Anweisung bewirkt das Zeichnen (bzw. Löschen) der Linie. Die x-Koordinaten x1,x2 der Linienpunkte müssen zwischen 0 und 639 liegen; die y-Koordinaten y1,y2 zwischen 0 und 399, also im „Bildschirm-Koordinatensystem“.

Bevor wir nun ein Testprogramm zum Zeichnen von Linien schreiben wollen, zunächst etwas über den Bildschirm des CPC 464. Es gibt 3 verschiedene Darstellungsarten, von denen nur jeweils eine aktiv sein kann; d. h., daß sie nicht gemischt werden können. Diese 3 Darstellungsarten sind:

MODE 0	MODE 1	MODE 2
Vielfarbmodus	Normalmodus	Hochauflösend
20 Spalten u.	40 Spalten u.	80 Spalten u.
25 Zeilen für Text	25 Zeilen für Text	25 Zeilen für Text
160x200 Punkte	320x200 Punkte	640x200 Punkte
16 Farben darstellbar	4 Farben darstellbar	2 Farben darstellbar

Diese Darstellungsarten werden durch die MODE-Anweisung eingeschaltet. Die Anweisung MODE 2 schaltet also den hochauflösenden Bildschirm ein. Darüber hinaus gibt es natürlich noch weitere Grafik-Befehle. Der einfachste lautet

```
BORDER n
```

Dabei steht n für eine Farbnummer, die aus folgender Palette ausgewählt werden kann.

0	Schwarz	7	Purpur	14	Pastellblau	21	Limonengrün
1	Blau	8	Hellmagenta	15	Orange	22	Pastellgrün
2	Hellblau	9	Grün	16	Rosa	23	Pastellcyan
3	Rot	10	Cyan	17	Pastellmagenta	24	Hellgelb
4	Magenta	11	Himmelblau	18	Hellgrün	25	Pastellgelb
5	Hellviolett	12	Gelb	19	Seegrün	26	Leuchtendweiß
6	Hellrot	13	Weiß	20	Hellcyan		

Bei dem Grünmonitor bezeichnen die Nummern unterschiedlich helle Grüntöne, wobei die Helligkeit mit der Größe der Zahl zunimmt. Die BORDER-Anweisung bezieht sich auf den nichtbeschreibbaren Bildschirmrahmen und füllt diesen mit der durch n bestimmten Farbe. Z. B. bewirkt BORDER 3 das Einschalten der Rahmenfarbe rot (3). Die BORDER-Anweisung arbeitet unabhängig vom Bildschirmmodus. Ein MODE-Kommando hat keine Auswirkung auf diesen Befehl.

Die Anzahl der gleichzeitig verwendbaren Farben für Grafik oder Text hängt vom Bildschirmmodus ab. Um farbige Grafiken zu erstellen, wird die Nummer des Farbstiftes angegeben, der verwendet werden soll. Diesem Farbstift ist eine bestimmte Farbnummer zugeordnet. Dabei muß zwischen Farbstift und Farbnummer unterschieden werden. Die Anweisung

```
DRAW x, y, m
```

zeichnet eine Linie von der augenblicklichen Cursorposition zu dem Punkt mit den Koordinaten (x,y), in der dem Farbstift m zugeordneten Farbe. Das m gibt also über den Farbstift die Farbe an. Von diesen Farbstiften können im Vielfarbmodus 16, im Normalmodus 4, und im hochauflösenden Modus 2 verschiedene verwendet werden. Diese Farbstifte sind nach dem Einschalten des Rechners mit Standardwerten vorbesetzt. Um diese nach eigenen Wünschen zu ändern benötigt man das INK-Kommando:

```
INK m, n
```

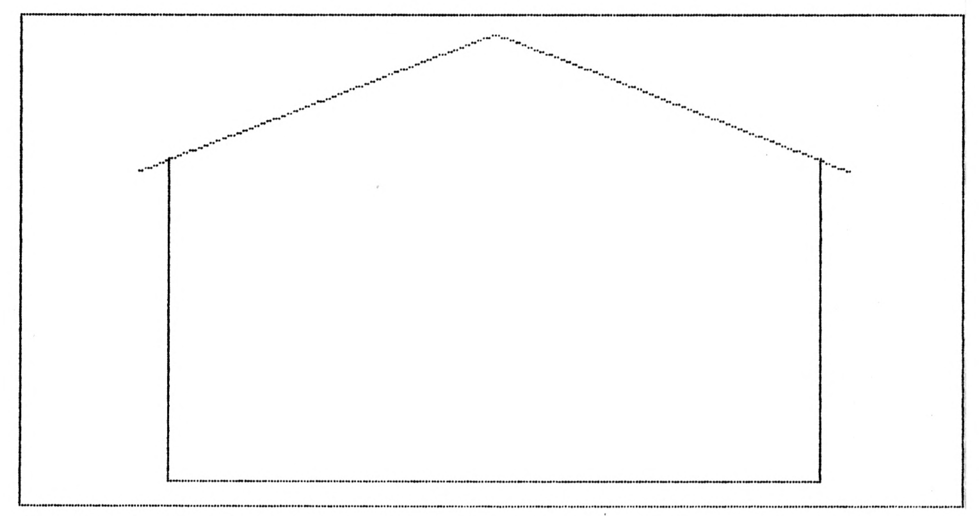
Durch diese Anweisung wird dem Farbstift mit der Nummer m die Farbe mit der Nummer n zugeordnet. Die Anzahl der INK's hängt vom Bildschirmmodus ab. Im hochauflösenden Modus können zwei Farbstifte (INK's) verwendet werden, nämlich INK 0 und INK 1. Im Vielfarbmodus können 16 Farbstifte verwendet werden, also INK 0 bis INK 15. Um die Textausgabe farbig zu gestalten, werden die Anweisungen

`PEN m` und `PAPER m`

benutzt . Nach dem PEN-Kommando werden die Zeichen auf dem Bildschirm in der dem Farbstift m zugeordneten Farbe dargestellt. Die PAPER-Anweisung bezieht sich auf den Hintergrund der Zeichen, der nach diesem Kommando in der Farbe dargestellt wird, die dem Farbstift m zugeordnet ist.

Wir wollen nun ein Programm zum Zeichnen von 5 Linien schreiben. In der Zeile 110 wird mittels der MODE-Anweisung der hochauflösende Bildschirm mit 640x200 Bildpunkten eingeschaltet. Danach wird der Bildschirmrahmen durch die Anweisung BORDER 2 in hellblau (auf dem Grünmonitor in einem dunklen Grünton) dargestellt. Mit der folgenden Anweisung INK 0,1 wird dem Farbstift mit der Nummer 0 die Farbe mit der Nummer 1 zugewiesen, d. h. ab jetzt besitzt der Farbstift 0 die Farbe blau. Entsprechend wird mit INK 1,26 dem Farbstift 1 die Farbe weiß (26) zugeordnet. Die Zahl am Ende der DRAW-Anweisungen gibt an, mit welchem Farbstift die Linien gezeichnet werden. Die Linien werden also in weiß (26) auf blauem (1) Bildschirm-Hintergrund gezeichnet. Die Warteschleife (Zeile 180) bewirkt, daß die Grafik so lange auf dem Bildschirm erhalten bleibt, bis irgendeine Taste gedrückt wird.

```
100 REM--- Linien zeichnen ---
105 :
110 MODE 2: BORDER 2 :INK 0,1:INK 1,26
115 :
120 MOVE 100, 20: DRAW 100,280, 1
130 MOVE  80,270: DRAW 320,380, 1
140 DRAW 560,270, 1
150 MOVE 540,280: DRAW 540, 20, 1
160 DRAW 100, 20, 1
170 :
180 a$=INKEY$: IF a$="" THEN 180
```



Grafik 1: Zeichnen von Linien

2.1 Zeichnen von dicken Linien

Das Hervorheben von Linien durch eine breitere Strichstärke ist bei technischen und künstlerischen Anwendungen oft notwendig. Auf einem Rasterbildschirm entsteht eine dickere Linie dadurch, daß mehrere Linien gezeichnet werden, die parallel zueinander versetzt sind.

Wir wollen nun ein Programm zum Zeichnen von dicken Linien entwickeln. Hierzu betrachten wir die Abb. 1. Die Linie vom Punkt P1(x1,y1) zum Punkt P2(x2,y2) soll dadurch dicker erscheinen, daß zusätzlich parallel versetzte Linien von Q1 nach Q2 gezeichnet werden. Die Koordinaten von Q1 sind durch x1−dx und y1+dx gegeben. Aus der Ähnlichkeit der beiden Dreiecke folgt:

```
dx = (y2-y1)/h
dy = (x2-x1)/h
```

Hierbei ist h nach dem Satz von Pythagoras durch

```
h = sqr ((x2-x1)*(x2-x1) +
    (y2-y1)*(y2-y1))
```

gegeben. Soll nach dem Zeichnen der Linie von P1(x1,y1) nach P2(x2,y2) zusätzlich die Linie von Q1 nach Q2 gezeichnet werden, so ist h und nachfolgend dx und dy zu berechnen. Analog ergeben sich die Koordinaten von

```
Q2(x2-dx,y2+dy)
```

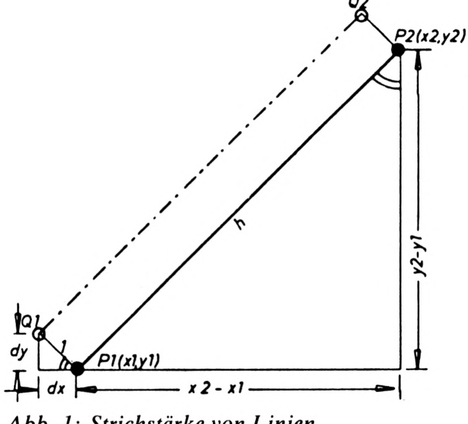


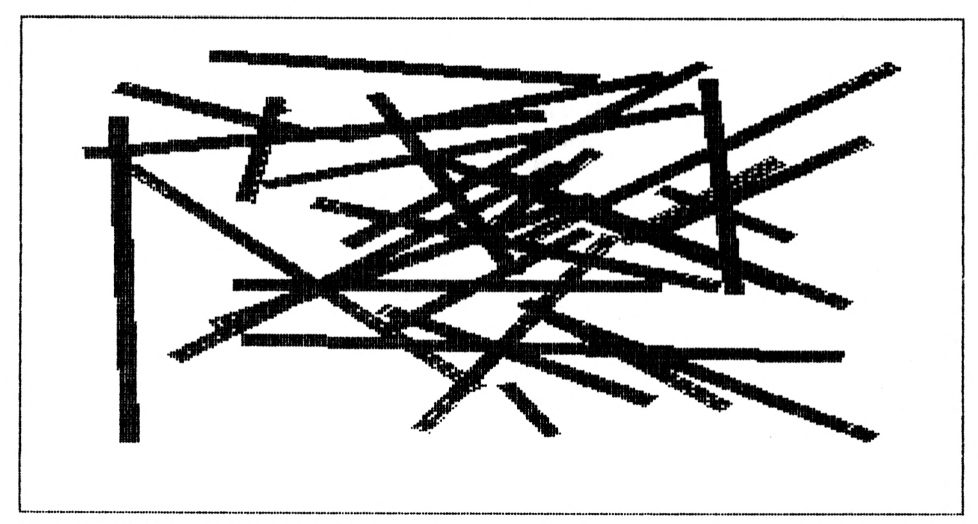
Abb. 1: Strichstärke von Linien

Das Programm

```
520 REM---  Dicke Linien zeichnen ---
525 :
530 MODE 2: BORDER 11:INK 0,0:INK 1,26
535 :
540 FOR k=1 TO 30
550 :   x1 = 40 + 560*RND(1): y1 = 30 + 340*RND(1)
560 :   x2 = 40 + 560*RND(1): y2 = 30 + 340*RND(1)
565 :
570 :   d = 10
580 :   h = SQR((x2-x1)^2 + 2.56*(y2-y1)^2)
590 :   hx = (x2-x1)/h/2.56: hy = (y2-y1)/h
595 :
600 :     FOR t=0 TO d STEP 1
610 :     dx = t*hy: dy = t*hx
620 :     MOVE x1-dx, y1+dy: DRAW x2-dx, y2+dy, 1
630 :     MOVE x2+dx, y2-dy: DRAW x1+dx, y1-dy, 1
640 :     NEXT t
645 :
650 NEXT k
655 :
660 a$=INKEY$ : IF a$="" THEN 660
```

folgt den erläuterten Darlegungen. In der Zeile 530 wird durch den MODE-Befehl der hochauflösende Bildschirm eingeschaltet. Mit den INK-Anweisungen werden die Farben festgelegt. Es werden weiße Linien (26) auf schwarzem Hintergrund (0) gezeichnet. Der Bildschirmrahmen wird mit BORDER 11 in himmelblau dargestellt. Wir wollen testweise 30 dicke Linien zeichnen. Die Zeile 540 zählt die Anzahl der dicken Linien. Der Anfangs-

punkt P1(x1,y1) und Endpunkt P2(x2,y2) jeder Linie wird zufällig (Zeilen 550,560) mit Hilfe der RND(1)-Funktion gewählt.



Grafik 2: Dick gezeichnete Zufallslinien

In der Zeile 580 wird h berechnet. Der Faktor 1.6*1.6=2.56 berücksichtigt die unterschiedliche Anzahl von Pixels (639x399) in x- bzw. y-Richtung. In der Zeile 590 wird hx,hy berechnet. Durch die Schleife von Zeilennummer 600 bis 640 werden nebeneinander Linien gezeichnet, die parallel zur Linie von P1(x1,y1) nach P2(x2,y2) sind. Hierbei entspricht d (Zeilen 570,600) der Anzahl der parallelen Linien und damit der Liniendicke (Strichstärke).

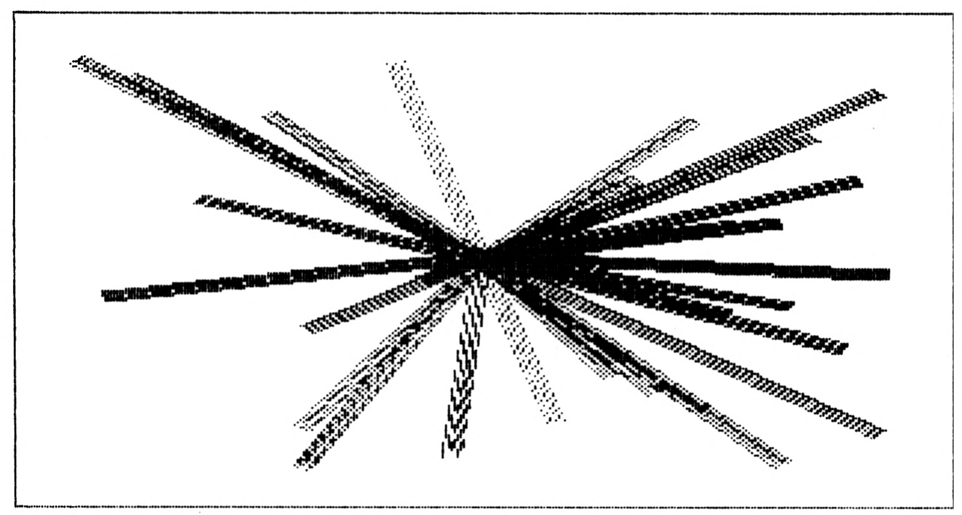
Die Grafik 2 zeigt dicke Zufallslinien (d=10). Jeder neue Programmstart (RUN) erzeugt eine andere Anordnung der 30 Zufallslinien. Setzen wir in der Zeile 570 d=0, so würde jeweils eine Linie von P1 nach P2 gezogen. Für d=10 werden 11 parallele Linien gezeichnet, die bei der Schrittweite 1 (Zeile 600) dicht nebeneinander liegen. In der Zeile 550 wird x1 mit einer Zufallszahl zwischen 40 und 600 und y1 mit einer Zufallszahl zwischen 30 und 370 besetzt. Entsprechendes gilt für x2,y2 (Zeile 560). Diese Zufalls-Anfangspunkte und Zufalls-Endpunkte werden durch dicke Linien verbunden.

Die k-Schleife (Zeile 540) zählt lediglich die Anzahl der Zufallslinien. Ersetzen wir die Programmzeile 570 durch

```
570 d = 10*RND(1)
```

so wird jede Linie mit einer zufälligen Liniendicke gezeichnet. Auf die Abbildung einer entsprechenden Grafik wollen wir verzichten.

Die Grafik 3 entspricht dick gezeichneten Zufallslinien mit einer größeren Schrittweite (Zeile 600). Als Schrittweite wurde STEP 5 gewählt. Die Grafik enthält einen zentralen



Grafik 3: Dick gezeichnete Linien mit einem zentralen Punkt

Punkt und wurde durch das Ersetzen der Zeile 550 durch die festen P1-Koordinaten

```
550 x1 = 320: y1 = 200
```

erzeugt.

2.2 Zeichnen von gestrichelten Linien

Bei kommerziellen Grafiksystemen ist es möglich, die Linienart zu wählen. In einfacher Weise ermöglichen solche Systeme das Zeichnen von gestrichelten, strichpunktierten oder gepunkteten Linien. Im Schneider-BASIC sind keine Linienarten vorgesehen. Daher soll hier ein Programm zum Zeichnen einer gestrichelten Linie vom Punkt P1(x1,y1) zum Punkt P2(x2,y2) angegeben werden:

```
720 REM--- Gestrichelte Linien zeichnen ---
725 :
730 MODE 2:BORDER 3:INK 0,0:INK 1,26
735 :
740 FOR k=1 TO 30
750 x1 = 40 + 560*RND(1): y1 = 30 + 340*RND(1)
752 REM---                x1 = x2: y1 = y2
760 x2 = 40 + 560*RND(1): y2 = 30 + 340*RND(1)
762 REM---                IF k=1 THEN 920
765 :
770 MOVE x1-4, y1+4: DRAW x1+4, y1+4, 1
775 DRAW x1+4, y1-4, 1: DRAW x1-4, y1-4,1
```

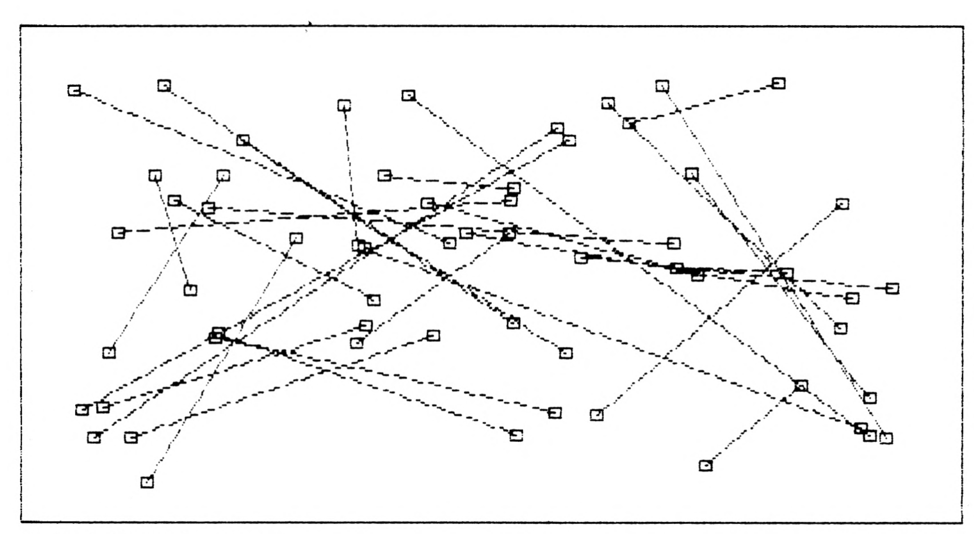
```
780 DRAW x1-4, y1+4, 1
790 MOVE x2-4, y2+4: DRAW x2+4, y2+4, 1
795 DRAW x2+4, y2-4, 1: DRAW x2-4, y2-4, 1
800 DRAW x2-4, y2+4, 1
805 :
810 :   h1 = 3: REM---          h1 = 1 + 4*RND(1)
820 :   h = SQR((x2-x1)^2 + (y2-y1)^2)/4
830 :   hx = (x2-x1)/h: hy = (y2-y1)/h
840 :   xa = x1: ya = y1: IF h<h1 THEN 910
850 :
860 :     FOR t=h1 TO h-h1 STEP h1
870 :     xb = x1 + t*hx: yb = y1 + t*hy
880 :     MOVE xa, ya: DRAW xb, yb, 1
890 :     xa = xb + hx: ya = yb + hy
900 :     NEXT t
905 :
910 :   MOVE xa, ya: DRAW x2, y2, 1
915 :
920 NEXT k
925 :
930 a$=INKEY$ : IF a$="" THEN 930
```

Das Programmstück von Zeile 810 bis 910 zeichnet eine gestrichelte Linie. Die anderen Programmzeilen dienen lediglich dem Austesten des Programmes mit Zufallslinien (Zeile 750,760). Der Wert h1=3 (Zeile 810) entspricht der Länge der Teilstriche. Für h1 können z. B. Werte zwischen 1 und 10 gewählt werden. Die Längen der Lücken können durch den Nenner (hier 4) der Zeile 820 verändert werden. Wird statt des Nenners 4 der Nenner 2 gewählt, so werden die Lücken verkürzt. In der Zeile 870 wird gemäß der Parameterdarstellung von Geraden (s. 5.1) der Endpunkt (xb,yb) eines Teilstrichelchens berechnet. In der Zeile 880 wird die Teilstrecke von (xa,ya) nach (xb,yb) auf der Verbindungslinie von (x1,y1) nach (x2,y2) gezeichnet. Die Zeile 890 bewirkt, daß der Anfangspunkt (xa,ya) der nächsten Teilstrecke weitergeschoben wird. Dadurch entsteht eine Lücke. Die Lückenlänge ist durch hx und hy festgelegt (Zeilen 820,830). Durch die Zeile 910 wird das letzte Teilstück der Strecke von (x1,y1) nach (x2,y2) gezeichnet.

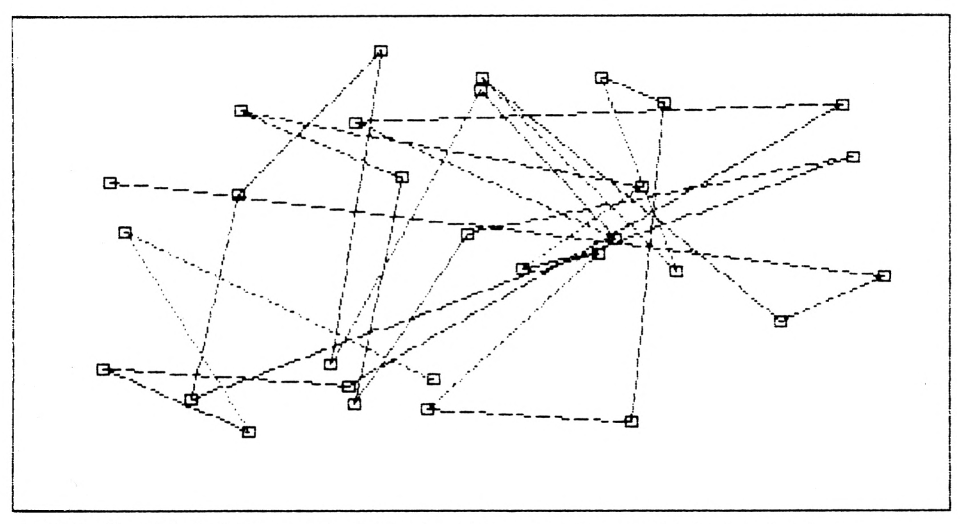
Die Zeilen 810 bis 910 können leicht als Unterprogramm formuliert werden, wobei x1,y1, x2,y2 Eingangswerte darstellen. Im Unterprogramm werden die internen Variablen t,h,h1,hx,hy,xa,ya,xb,yb verwendet.

Das beschriebene Programm liefert die Grafik 4. Die Anfangspunkte (x1,y1) und Endpunkte (x2,y2) der gestrichelten Linien sind durch Rechtecke (Zeilen 770 bis 800) markiert worden. Die Werte x1,y1, x2,y2 ergeben sich als Zufallszahlen (Zeilen 750,760). Werden die durch

```
REM---
```



Grafik 4: Gestrichelte Linien



Grafik 5: Gestrichelte Linien als fortlaufender Linienzug mit unterschiedlichen Stricharten

gekennzeichneten Zeilen aktiviert, so wird der Linien-Endpunkt als neuer Linien-Anfangspunkt verwendet (Zeile 752). Durch die zufällige Wahl von h1 (Zeile 810) zwischen 1 und 5 ergeben sich unterschiedliche Strichelungen, die in der Grafik 5 dargestellt sind.

3. Bildschirmkoordinaten

3.1 Die Skalengleichung

Die Abmessungen eines darzustellenden Objektes sind im allgemeinen verschieden von den Bildschirmabmessungen. Zur grafischen Darstellung auf dem Bildschirm sind die Objektabmessungen in Bildschirmkoordinaten umzurechnen. Dies geschieht mit Hilfe eines Maßstabfaktors.

Wir wollen den Umrechnungsfaktor mit dem Strahlensatz ermitteln. Im Punkt Q der Ab. 2 sei z. B. eine Lichtquelle angeordnet. Ein Lichtstrahl, der von der Lichtquelle Q zum Objektpunkt R verläuft, markiert auf der i-Achse die Schattengrenze S. Die Schattenlänge hängt von der Objektlänge ab. Durch die Differenz (x−x8) der x-Achsenwerte ist die Objektlänge gegeben. Die Länge des Schattens entspricht (i−i8). Eine größere Objektlänge ergibt eine größere Schattenlänge. Nach dem Strahlensatz ist das Verhältnis der Schattenlänge zur Objektlänge konstant. Diese Konstante bezeichnen wir als x-Maßstabsfaktor. Durch Formeln ausgedrückt ergibt sich:

$$\frac{i-i8}{x-x8} = \frac{i9-i8}{x9-x8} = mx$$

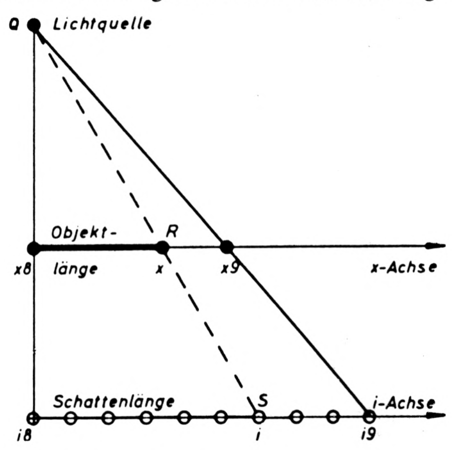


Abb. 2: Strahlensatz

Hierbei ist x8 (bzw. x9) die minimale (bzw. maximale) x-Koordinate des abzubildenden Gegenstandes. Soll z. B. ein Haus grafisch dargestellt werden, so sind diese Abmessungen und somit x8 und x9 bekannt. Die i-Achse soll den waagerechten Bildpunkten (Pixels) einer Bildschirmzeile entsprechen.

Diese Bildschirmpunkte können vom i8-ten bis i9-ten Pixel gesetzt werden. Der CPC-464-Bildschirm hat in einer Zeile 0,1,2,. . .,639 ansteuerbare Pixels. Sollen nicht alle Pixels einer Zeile verwendet werden, so könnte z. B. i8=50, i9=550 gewählt werden. Wenn x von x8 bis x9 läuft, so würde der 50. bis 550. Bildpunkt angesprochen. Weil der o. g. Strahlensatz für viele grafische Darstellungen verwendet wird, wollen wir zum besseren Verständnis ein Beispiel per Hand durchrechnen. Die i-Achse soll vom 50. bis 550. Pixel einer Bildschirmzeile gehen, wenn die x-Werte von 5 bis 15 laufen. Wir erhalten den Maßstabsfaktor

$$mx = \frac{i9-i8}{x9-x8} = \frac{550-50}{15-5} = 50$$

Aus dem Strahlensatz ergibt sich der zu einem Wert gehörende i-te Pixel zu

```
i = i8 + mx*(x-x8)
i = 50 + 50*(x-5)
```

Wird x eingesetzt, so erhalten wir folgende Tabelle:

x	5.0	6.0	7.0	8.0	9.0	10.0	11.0	12.0	13.0	14.0	15.0
i	50.0	100.0	150.0	200.0	250.0	300.0	350.0	400.0	450.0	500.0	550.0

```
100 REM--- x8=5: x9=15: i8=50: i9=550:GOTO 115
105 INPUT " x8, x9 ="; x8, x9
110 INPUT " i8, i9 ="; i8, i9
112 :
115 i8 = 0.5 + INT(i8): i9 = 0.5 + INT(i9)
120 :
130 mx = (i9 - i8)/(x9 - x8)
135 :
145 FOR x=x8 TO x9 STEP (x9-x8)/10
155 :   i% = i8 + mx * (x - x8)
170 :   PRINT x, i%
180 NEXT x
```

Die i-Werte müssen auf ganze Zahlen gerundet werden, weil es nur ganzzahlige Pixelwerte gibt. Im obigen Programm wird durch die Addition von 0.5 zu i8 und i9 (Zeile 115) diese Rundung bewirkt. Die Nachkommastellen werden bei der Besetzung der Ganzzahlvariablen i% abgeschnitten (Zeile 155).

3.2 Zeichnen einer x-Achse

Ein Achsenkreuz wird u. a. für die grafische Darstellung von Funktionen benötigt. Eine Achse besteht aus einer mit Markierungen versehenen Linie. Das folgende Programm folgt den Ausführungen des vorherigen Kapitels über die Skalengleichung. In der Zeile 230 wird der Maßstabsfaktor mx berechnet. Die Umrechnung der x-Werte in die Pixelnummer i% wird durch die Formel in der Zeile 250 durchgeführt. Zum Abschneiden der Nachkommastellen wird die Ganzzahlvariable i% verwendet. Auch möglich wäre die Verwendung der INT()-Funktion. In der Zeile 265 wird eine waagerechte Linie vom „alten“ Pixel i1 zum nachfolgenden Pixel i% gezeichnet. Die Zeile 260 zeichnet einen Querstrich beim alten Pixel und markiert damit die x-Achse. Die Zeilen 270 und 275 beschriften die markierte „alte“ Stelle x1 der x-Achse mit dem zugehörigen x1-Wert.

```
200 REM--- x-Achse zeichnen und beschriften ---
210 x8 = 5: x9 = 15: i8 = 50.5: i9 = 550.5
220 j = 300:MODE 1:BORDER 3:INK 0,1:INK 1,24
225 :
230 mx = (i9 - i8)/(x9 - x8)
240 :
245 :   FOR x=x8 TO x9 STEP (x9-x8)/10
250 :   i% = i8 + mx * (x-x8)
255 :   x1$ = STR$(x1): IF x=x8 THEN 280
260 :   MOVE i1, j : DRAW i1, j-10, 1
265 :   MOVE i1, j : DRAW i%,  j , 1
270 :   TAG: MOVE i1-10*LEN(x1$), j-25
275 :   PRINT x1$;
280 :   i1 = i% : x1 = x
285 :   NEXT x
290 :
300 r = 6*SGN(x9-x8)
305 MOVE i%-5*r, j-r: DRAW  i% , j , 1
310 DRAW i%-5*r, j+r, 1: DRAW i%-5*r, j-r, 1
320 :
330 h$="AEQUIDISTANTE  ACHSENEINTEILUNG"
340 MOVE 50, j-60: PRINT h$;
350 TAGOFF
360 :
370 a$=INKEY$ : IF a$="" THEN 370
```

Die Zeilen 305 und 310 zeichnen den Richtungspfeil an die x-Achse. Infolge der Zeile 300 zeigt dieser Pfeil in die Richtung der wachsenden x-Werte. Die Grafik 6 enthält die x-Achse mit äquidistanter Einteilung.

Auf das Zeichnen einer y-Achse kann hier verzichtet werden. Analog zu den obigen Betrachtungen kann leicht ein Programm zum Zeichnen einer y-Achse aufgestellt werden.

3.3 Zeichnen von Funktionsskalen

Bisher betrachteten wir Achsen mit äquidistanten Einteilungen. Diese gleichmäßige Einteilung ist für manche Anwendungen weniger geeignet. Z. B. werden Schallabsorptionsmessungen i. allg. über einer logarithmisch eingeteilten Frequenzskala aufgetragen. Hierfür kann z. B. „Logarithmus-Papier“ verwendet werden. Die Abstände auf der x-Achse entsprechen dem log(x). In der Nomographie werden Funktionsskalen in vielfacher Weise angewendet. Wir wollen solche Funktionsskalen vom Computer zeichnen lassen. Hierzu dient das folgende Programm, das nur geringfügig gegenüber dem vorhergehenden Programm zum Zeichnen einer Achse verändert wurde.

```
400 REM--- log(x)-Achse zeichnen ---
405 :
410 x8 = 5: x9 =15: i8 = 50.5: i9 =550.5: j =120
415 :
420 DEF FN f(x) = LOG(x)
430 mx = (i9-i8)/( FN f(x9) - FN f(x8) )
435 MODE 1: BORDER 3:INK 0,1:INK 1,24
437 TAG
440 :
445 :   FOR x=x8 TO x9 STEP (x9-x8)/10
450 :   i% = i8 + mx * (FN f(x) - FN f(x8))
455 :   IF x=x8 THEN 485
460 :   IF i%-il<10 THEN 490
465 :   MOVE il, j-10
470 :   DRAW il, j, 1: DRAW i%, j , 1
472 :
475 :   xl$ = STR$(xl): i2 = 10*LEN(xl$)
480 IF i%-i3>i2 AND i2<31
    THEN MOVE il-i2,j-25: PRINT xl$;: i3=il+i2+25
485 :   il = i%: xl = x
490 :   NEXT x
492 :
495 r = 6*SGN(x9-x8)
500 MOVE i%-5*r, j+r: DRAW i%, j , 1
505 DRAW i%-5*r, j-r, 1: DRAW i%-5*r, j+r, 1
520 :
610 h$="LOGARITHMISCHE  ACHSENEINTEILUNG"
620 MOVE 50, 60: PRINT h$;
630 TAGOFF
640 :
650 a$=INKEY$: IF a$="" THEN 650
```

In der Zeile 420 wird die Skalenfunktion f(x) definiert. Wenn wir eine reziprok eingeteilte Skala erzeugen wollten, so wäre die Zeile

```
420 DEF FN f(x) = 1/x
```

zu verwenden. Die gleiche Achse mit einer äquidistanten Einteilung wird gezeichnet für

```
420 DEF FN f(x) = x
```

oder z. B. für

```
420 DEF FN f(x) = 2 * x + 3
```

Da wir eine logarithmisch geteilte Skala zeichnen lassen wollen, verwenden wir

```
420 DEF FN f(x) = LOG(x)
```

Die x-Werte sollen von 5 bis 15 laufen (Zeile 410). In der Zeile 430 wird der Maßstabsfaktor

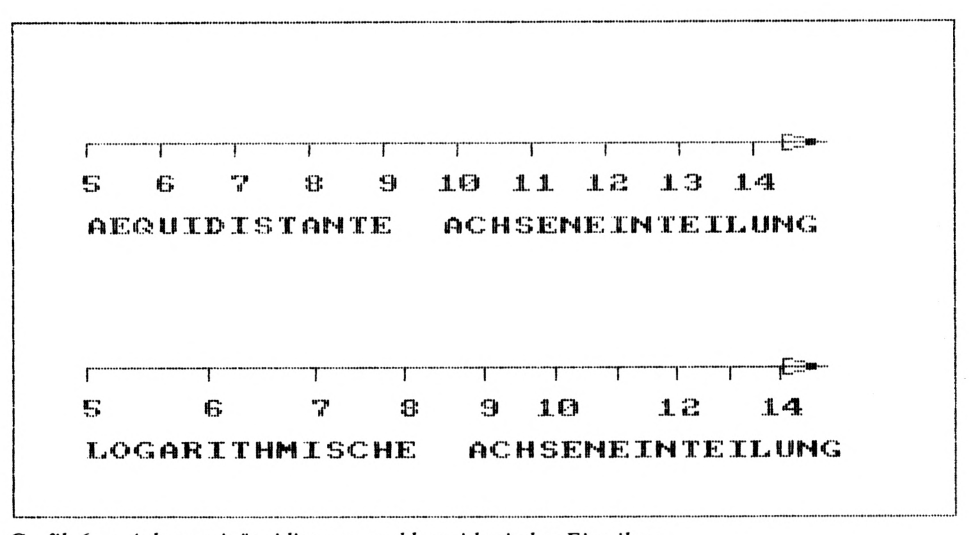
$$mx = \frac{i9-i8}{f(x9)-f)x8)}$$

berechnet. Die Umrechnung der x-Werte in die entsprechenden Pixels geschieht durch

```
i% = i8 + mx * ( f(x) - f(x8) )
```

In der Zeile 450 wird diese Umrechnung durchgeführt. Liegen die Achsenmarkierungen zu dicht beieinander, so werden durch die Zeile 460 Markierungsstrichelchen unterdrückt.

Die Markierungen werden nur dann beschriftet, wenn keine Überdeckung der Ziffern auftritt (siehe Zeile 480). In der Grafik 6 ist eine x-Achse mit logarithmischer Einteilung enthalten.



Grafik 6: x-Achsen mit äquidistanter und logarithmischer Einteilung

3.4 Berechnung der Bildschirmkoordinaten

Der Rasterbildschirm besteht aus einer bestimmten Anzahl ansteuerbarer Bildpunkte, die einzeln „gesetzt“ werden können. Dieses Setzen eines Punktes geschieht durch die Angabe

der entsprechenden Zeile und Spalte des hochauflösenden Bildschirms. Dadurch ist es möglich, die Lage eines Punktes durch Angabe der Zeile i und Spalte j zu beschreiben. Bei dieser Numerierung der Punkte kann i von i8(min) bis i9(max) und j von j8(min) bis j9(max) laufen. Zum Beispiel könnte i von 0 bis 639 und j von 0 bis 399 reichen. Da der CPC 464 in der senkrechten Richtung jeweils zwei Pixel zu einem zusammenfaßt, ergeben sich 200 Bildpunkte, obwohl j von 0 bis 399 reicht. Diese 640x200=128 000 Bildpunkte bilden das physikalische Koordinatensystem. Der Punkte (0,0) liegt in der linken unteren Ecke des Bildschirmes. Die darzustellenden Kurven, Figuren oder Punkte variieren gewöhnlich nicht zwischen i8 und i9 bzw. j8 und j9, sondern zwischen den Werten x8 und x9 bzw. y8 und y9. Die Umrechnung dieser Koordinaten des mathematischen x,y-Koordinatensystems auf die entsprechenden i,j-Bildpunkte des physikalischen Bildschirmsystems geschieht mit dem Strahlensatz

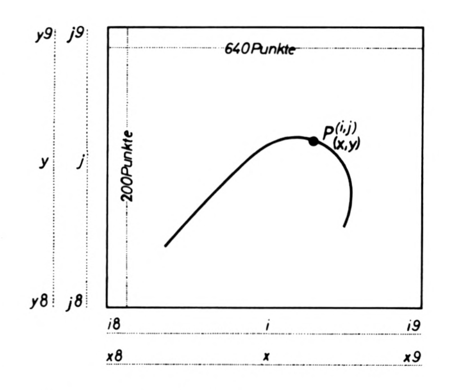


Abb. 3: Rasterbildschirm

```
mx = (i9-i8)/(x9-x8) = (i-i8)/(x-x8)
my = (j9-j8)/(y9-y8) = (j-j8)/(y-y8)
```

der ausführlich im Kapitel 3.1 beschrieben ist.

Die Umstellung dieser Formeln ergibt:

```
i = i8 + mx * (x-x8)
j = j8 + my * (y-y8)
```

Der kleinste x-Wert (x8) und der größte x-Wert (x9) des abzubildenden Gegenstandes ist im allgemeinen bekannt. Ebenso die minimalen und maximalen y-Werte y8 und y9. Mit i8=0.5,i9=639.5 und j8=0.5,j9=399.5 können die Maßstabsfaktoren mx=(i9-i8)/(x9−x8),my=(j9−j8)/(y9−y8) berechnet und gespeichert werden. Für alle Umrechnungen i=i8+mx*(x−x8),j=j8+my*(y−y8) von Zwischenpunkten (x,y) in die Bildschirmkoordinaten werden dann diese mx-, my-Werte verwendet. Mit den physikalischen Koordinaten i,j kann der zu x,y gehörende Bildpunkt (i,j) angesprochen werden. Da es jedoch nur ganzzahlige Bildschirmpunkte gibt, ist von i,j nur der ganzzahlige Anteil relevant. Für diese Operation bieten sich die Ganzzahlspeicher i%,j% oder die INT()-Funktion an.

Es gilt:

```
x8 =< x <= x9 und y8 =< y <= y9
i8 =< i <= i9 und j8 =< j <= j9
```

4. Grafische Bildschirmfenster

Das Schneider-BASIC bietet die Möglichkeit, Bildschirmfenster zu definieren, wobei zwischen Text- und Grafikfenstern unterschieden wird. An dieser Stelle wird nur auf die Grafikfenster eingegangen. Um einen Bildschirmteilbereich für die grafische Darstellung zu nutzen, müssen wir diesen mit Hilfe der Anweisung

```
ORIGIN x0,y0, l, r, o, u
```

festlegen. Dabei bestimmen l und r die Koordinaten der linken bzw. rechten Ecke des Bildschirmteilbereiches. Mit o und u werden die Koordinaten der oberen bzw. unteren Grenze festgelegt. Mit diesen 4 Angaben wird also ein Bildschirmteilbereich bestimmt, dessen eigene Koordinaten der linken unteren Ecke durch x0,y0 festgelegt werden. Für das Bildschirmfenster der Abb. 4 ergeben sich durch die Wahl von i8=200,i9=540,j8=120,j9=320 die Maßstabsfaktoren mx=340/(x9−x8),my=200/(y9−y8) und damit die Abbildungsgleichungen

```
i% = i8 + mx * (x-x8)
j% = j8 + my * (y-y8)
```

Für x-Werte zwischen x8 und x9 und y-Werte zwischen y8 und y9 ergeben sich Punkte (i%,j%) auf dem Bildschirm, die im gewählten Bildschirmfenster liegen.

Das folgende Programm zeigt die Verwendung eines Bildschirmfensters. In der Zeile 110 wird das Bildschirmfenster nach Lage und Größe gewählt. Die minimalen und maximalen x- und y-Werte sind in der Zeile 120 festgelegt. Das gewählte Bildschirmfenster wird in der Zeile 150, nach dem Einschalten des hochauflösenden Bildschirmes, durch

```
ORIGIN 0,0,i8,i9,j9,j8
```

bestimmt und durch ein Rechteck (Zeilen 160,170,180) sichtbar begrenzt.

Die Zeilen 240 bis 270 rechnen die gelesenen Endpunkte P1(x1,y1) und P2(x2,y2) einer Strecke in die physikalischen Koordinaten (i1%,j1%) und (i2%,j2%) des Bildschirmfensters um. Die Zeile 280 veranlaßt das Zeichnen der entsprechenden Linie. Liegen Teile dieser Linie außerhalb des Fensters, so wird das Zeichnen dieser außerhalb liegenden Punkte automatisch vom Rechner unterdrückt. Durch Veränderungen in den DATA-Zeilen 340 bis 360 können die Auswirkungen eines Programmlaufes beobachtet werden.

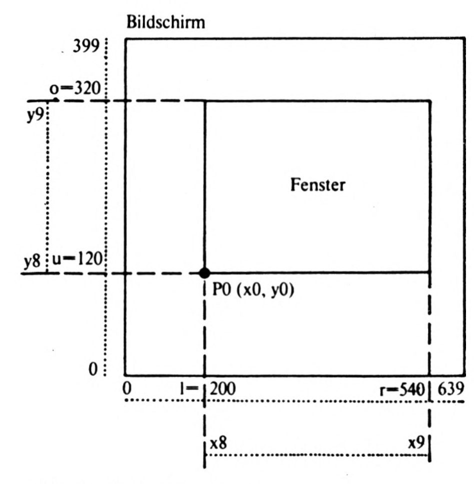
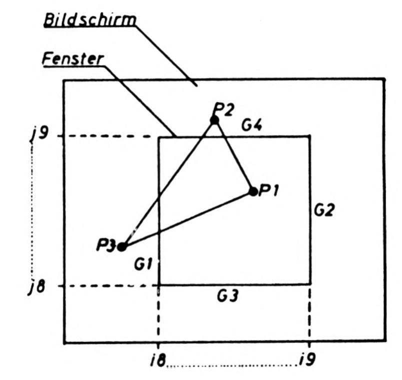


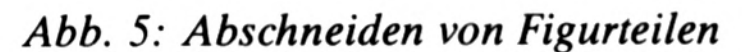
Abb. 4: Bildschirmfenster

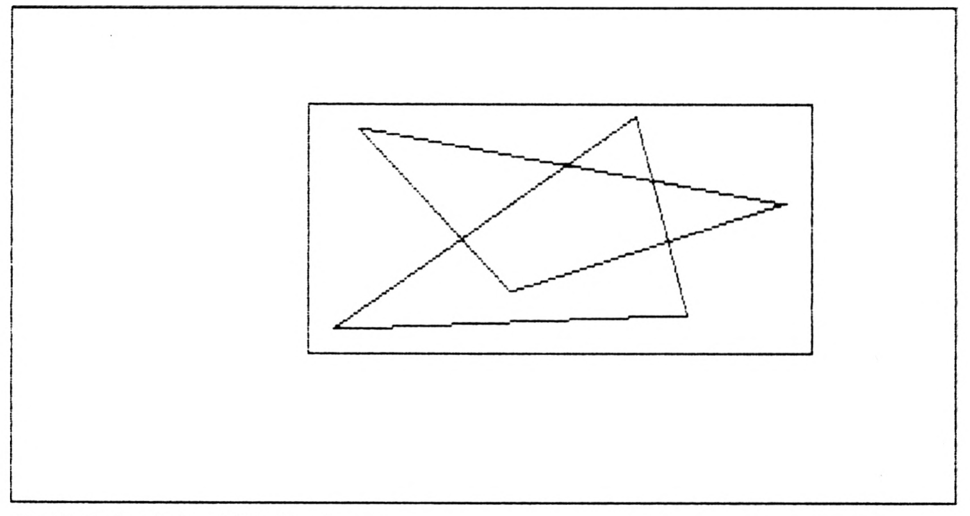
```
100 REM--- Bildschirmfenster ---
110 i8=200: i9=540: j8=120: j9=320
120 x8=-10: x9= 10: y8=-10: y9= 10
130 MODE 2: BORDER 3: INK 0,1: INK 1,24
140 :
150 ORIGIN 0,0,i8,i9,j9,j8
160 MOVE i8,j8:DRAW i9,j8,1
170 DRAW i9,j9,1:DRAW i8,j9,1
180 DRAW i8,j8,1
190 :
200 mx=(i9-i8)/(x9-x8): my=(j9-j8)/(y9-y8)
210 :
220 :   FOR k=1 TO 6
230 :   READ x1,y1, x2,y2
240 :   i1% = i8 + mx*(x1-x8)
250 :   j1% = j8 + my*(y1-y8)
260 :   i2% = i8 + mx*(x2-x8)
270 :   j2% = j8 + my*(y2-y8)
280 :   MOVE  i1%, j1% : DRAW  i2%, j2%, 1
290 :   NEXT k
300 :
310 a$=INKEY$: IF a$="" THEN 310: REM warten
320 END
330 :
340 DATA -8, 8,  -2,-5,  -2,-5,   9, 2
350 DATA  9, 2,  -8, 8,   3, 9,  -9,-8
360 DATA -9,-8,   5,-7,   5,-7,   3, 9
```

Es kann bei der Vergrößerung eines Bildausschnittes vorkommen, daß Teile der Figur außerhalb des gewählten Fensters liegen (s. Abb. 5). Die Unterdrückung dieser Teile wird vom Schneider-BASIC übernommen.

Das folgende Programm verdeutlicht die Unterdrückung außerhalb des Fensters liegender Punkte. Dabei wollen wir zufällig gezeichnete Linien verwenden. In der Zeile 200 wird das grafische Bildschirmfenster bestimmt, dessen Größe sich aus den gewählten Werten für i8,i9,j8,j9 ergibt (Zeile 140). Durch die Schleife von Zeile 250 bis 350 werden 40 zufällig ausgewählte Linien gezeichnet. Dabei werden außerhalb des begrenzten Fensters liegende Punkte




Abb. 5: Abschneiden von Figurteilen

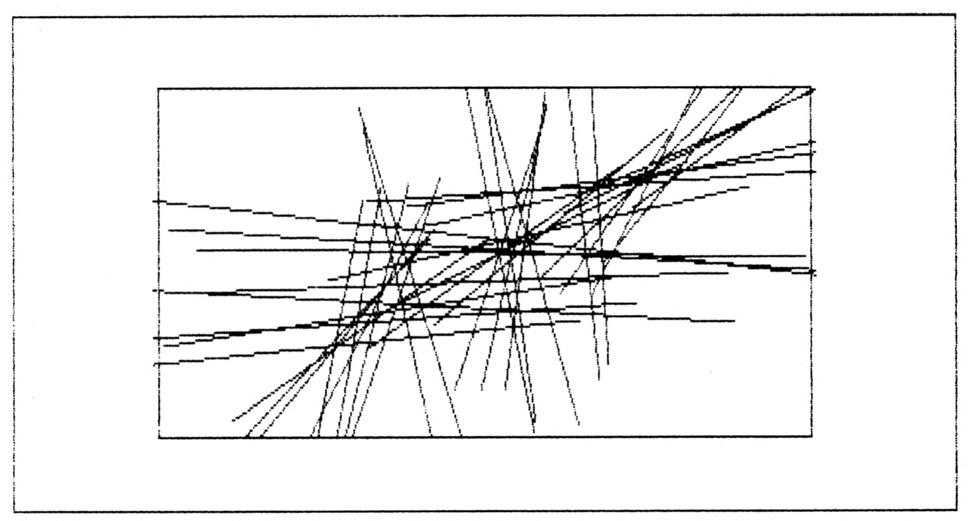


Grafik 7: Grafisches Bildschirmfenster

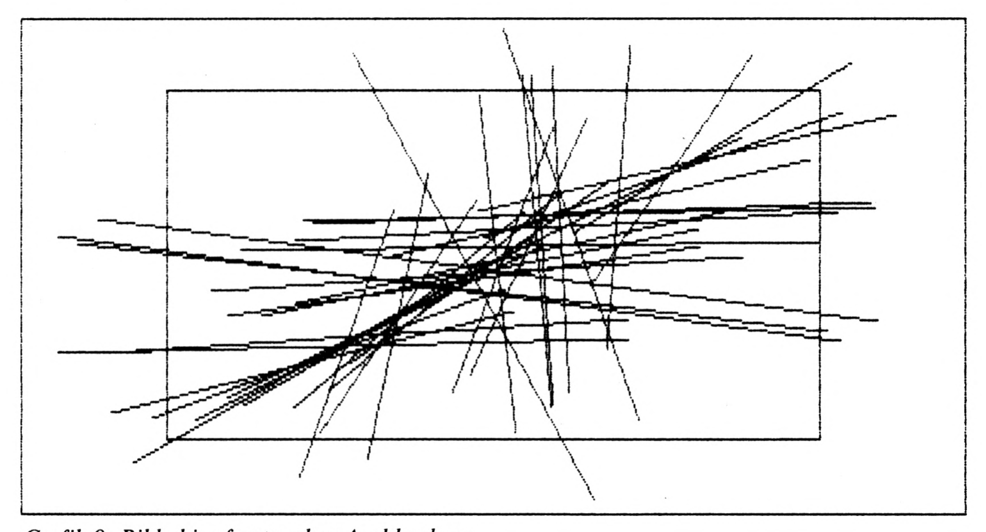
unterdrückt (s. Grafik 8). Dieses wird besonders deutlich, wenn, durch ein hinter die Zeilennummer eingeschobenes „REM“, die Zeile 200 deaktiviert wird. Das Fenster wird dann nicht mehr eingerichtet und alle Linien werden ohne Unterdrückung irgendwelcher Teile gezeichnet (s. Grafik 9).

```
120 REM---  Bildschirmfenster   ---
130 REM--- mit/ohne Ausblendung ---
140 i8=100: i9=540: j8= 60: j9=340
150 x8=  4: x9= 26: y8=  4: y9= 26
160 :
170 mx=(i9-i8)/(x9-x8): my=(j9-j8)/(y9-y8)
180 :
190 MODE 2:BORDER 3:INK 0,1:INK 1,24
200 ORIGIN 0,0,i8,i9,j9,j8
210 MOVE i8,j8: DRAW i9,j8,1
220 DRAW i9,j9,1: DRAW i8,j9,1
230 DRAW i8,j8,1
240 :
250 FOR n=1 TO 40
260 :
270 : x1 = 20*RND(1): y1 = 20*RND(1)
280 : x2 = y1 + 10: y2 = x1 + 10
290 : i1 = i8 + mx*(x1-x8)
300 : j1 = j8 + my*(y1-y8)
310 : i2 = i8 + mx*(x2-x8)
```

```
320 : j2 = j8 + my*(y2-y8)
330 : MOVE i1, j1: DRAW i2, j2, 1
340 :
350 NEXT n
360 :
370 a$=INKEY$: IF a$="" THEN 370
```



Grafik 8: Bildschirmfenster mit Ausblendung



Grafik 9: Bildschirmfenster ohne Ausblendung

5. Geradengleichung

5.1 Parameterdarstellung der Geraden

Bei der Programmierung von Computergrafiken wird die Parameterdarstellung von Geraden bevorzugt verwendet. Diese Darstellung

```
x = x1 + t * (x2-x1)
y = y1 + t * (y2-y1)
```

benutzt den Parameter zur Berechnung der x,y-Koordinaten von Punkten auf der Geraden. Wird z. B. die 1. und 2. Gleichung nach t umgestellt, so ergibt sich aus

$$t = \frac{x-x1}{x2-x1} = \frac{y-y1}{y2-y1}$$

die 2-Punkte-Form

$$\frac{y-y1}{x-x1} = \frac{y2-y1}{x2-x1}$$

der Geradengleichung. Soll eine Gerade durch die beiden Punkte P1(4,3),P2(7,6) gehen, so liefert die Parameterdarstellung der Geraden die beiden Gleichungen

```
x = 4 + 3*t
y = 3 + 3*t.
```

Durch das Programm:

```
500 REM--- Parameterdarstellung    ---
505 REM--- einer Geraden (Tabelle) ---
510 DATA  4, 3,   7, 6
520 READ x1,y1,  x2,y2
530 PRINT " t       x       y  "
540 PRINT "--------------------"
550 :
560 FOR t=-1.5 TO 2 STEP 0.5
570 x = x1 + t * ( x2 - x1 )
580 y = y1 + t * ( y2 - y1 )
590 PRINT t; TAB(7); x; TAB(14); y
600 NEXT t
```

wird eine Wertetabelle für die Gerade durch P1 und P2 erzeugt. Es ergeben sich die Werte

t	−1.5	−1.0	−0,5	0.0	0.5	1.0	1.5	2.0
x	−0.5	1.0	2.5	4.0	5.5	7.0	8.5	10.0
y	−1.5	0.0	1.5	3.0	4.5	6.0	7.5	9.0

Zu jedem t-Wert gehören jeweils x,y-Werte, die einen Geradenpunkt im x,y-System markieren (Abb. 6). Für t=0 ergibt sich der vorgegebene Geradenpunkt P1 und für t=1 ergibt sich P2.

Variiert t zwischen 0 und 1, so ergeben sich Geradenpunkte P(x,y), die zwischen P1 und P2 liegen. Soll eine Linie punktweise von P1 nach P2 gezeichnet werden, so läuft t von 0 bis 1 (Abb. 6). Der Parameter t ist negativ, wenn von P1 aus betrachtet die Punkte P(x,y) und P2(x2,y2) auf entgegengesetzten Seiten der Geraden liegen.

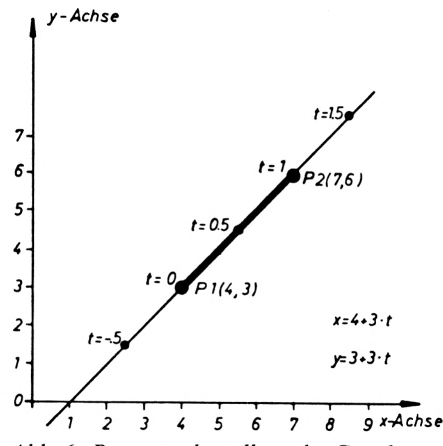


Abb. 6: Parameterdarstellung der Geraden

5.2 Schnittpunkt von zwei Geraden

Bei der Erstellung von Grafiken ist oft die Frage zu klären, ob eine von P1 nach P2 zu zeichnende Gerade auf der Zeichenstrecke einen Schnittpunkt mit einer anderen Geraden aufweist.

Zum Beispiel könnte die Linie von P1 nach P2 den Zeichenrand überschreiten und dürfte daher nur bis zum Zeichenrand geführt werden. Im folgenden soll der Schnittpunkt der beiden nicht parallelen Geraden G1 und G2 errechnet werden. Die beiden Geraden sind durch die Punkte P1(x1,y1), P2(x2,y2) bzw. Q1(u1,v1),Q2(u2,v2) festgelegt (s. Abb. 7). Für die vorgegebenen Punkte P1,P2 lautet die Parameterdarstellung der Geraden G1:

```
x = x1 + t * (x2-x1)
y = y1 + t * (y2-y1).
```

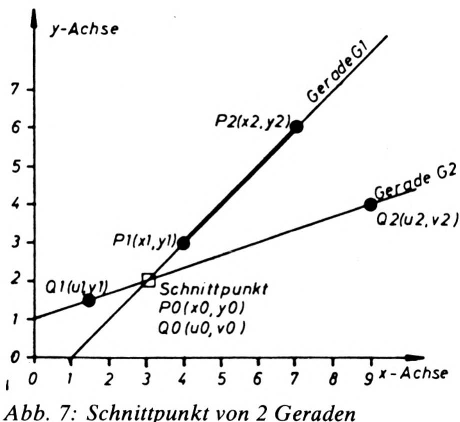


Abb. 7: Schnittpunkt von 2 Geraden

Für t-Werte zwischen 0 und 1 ergeben sich Geradenpunkte P(x,y), die zwischen P1 und P2 liegen. Mit Hilfe von t kann festgestellt werden, ob ein Geradenpunkt zwischen P1 und P2 liegt. In gleicher Weise gilt für die Gerade G2:

```
x = u1 + s * (u2-u1)
y = v1 + s * (v2-v1).
```

Für G2 wird der Parameter s verwendet. Nach der Umstellung der beiden Formeln ergibt sich:

$$s = \frac{x-u1}{u2-u1} = \frac{y-v1}{v2-v1}$$

Setzen wir hier x=x1+t0*(x2−x1) und y=y1+t0*(y2−y1) ein, so erhalten wir den Parameter t0 des Schnittpunktes von G1 und G2 zu:

$$s0 = \frac{(u1-x1) * (y2-y1) - (v1-y1) * (x2-x1)}{(x2-x1) * (v2-v1) - (y2-y1) * (u2-u1)}$$

Vorausgesetzt wird, daß die beiden Geraden G1,G2 nicht parallel sind, weil dann der Nenner in der t0-Formel Null wird. Mit dem berechneten t0-Wert ergeben sich die Schnittpunktkoordinaten zu:

```
x0 = x1 + t0 * (x2 - x1)
y0 = y1 + t0 * (y2 - y1).
```

In ähnlicher Weise kann der Schnittparameter für die Gerade G2 ermittelt werden:

$$t0 = \frac{(u1-x1) * (v2-v1) - (v1-y1) * (u2-u1)}{(x2-x1) * (v2-v1) - (y2-y1) * (u2-u1)}$$

Mit s0 ergeben sich die Schnittpunktkoordinaten u0=u1+s0*(u2−u1),v0=v1+s0*(v2−v1). Natürlich ist im Schnittpunkt u0=x0 und v0=y0. Die Nenner von t0 und s0 sind gleich. Der Schnittpunkt P0(x0,y0) von G1 und G2 liegt auf dem Geradenstück zwischen P1 und P2, wenn t0 zwischen 0 und 1 ist. Für einen s0-Wert zwischen 0 und 1 liegt der Schnittpunkt zwischen Q1 und Q2. Mit Hilfe von t0 (bzw. s0) können wir rechnerisch feststellen, ob der Schnittpunkt zwischen den Punkten P1 und P2 (bzw. Q1 und Q2) liegt.

In dem folgenden Programm wird der Schnittpunkt der Geraden G1 durch P1(4,3), P2(7,6) mit der Geraden G2 durch Q1(1.5,1.5),Q2(9,4) ermittelt und mit Hilfe der berechneten Parameter t0 und s0 darüber entschieden, ob ein Schnittpunkt zwischen P1 und P2 und/oder zwischen Q1 und Q2 vorliegt.

```
620 REM--- Schnittpunkt von 2 Geraden ---
630 REM--- in Parameterdarstellung    ---
640 DATA   4, 3,   7, 6,  1.5,1.5, 9, 4
650 READ  x1,y1,  x2,y2,   u1,v1, u2,v2
660 :
670 h  = (x2-x1)*(v2-v1) - (y2-y1)*(u2-u1)
680 h1 = (u1-x1)*(v2-v1) - (v1-y1)*(u2-u1)
690 h2 = (u1-x1)*(y2-y1) - (v1-y1)*(x2-x1)
695 :
700 IF h=0 THEN PRINT "g1 parallel zu g2": END
710 :
720 t0 = h1/h
730 x0 = x1 + t0*(x2-x1)
740 y0 = y1 + t0*(y2-y1)
750 s0 = h2/h
760 u0 = u1 + s0*(u2-u1)
770 v0 = v1 + s0*(v2-v1)
780 :
790 PRINT "Der Schnittpunkt  (";x0;",";y0;")  ";
800 IF t0>=0 AND t0<=1
    THEN PRINT "liegt zwischen p1 und p2"
810 IF t0< 0 OR  t0> 1
    THEN PRINT "liegt nicht zwischen p1 und p2"
815 :
820 PRINT "Der Schnittpunkt  (";u0;",";v0;")  ";
830 IF s0>=0 AND s0<=1
    THEN PRINT "liegt zwischen q1 und q2"
840 IF s0< 0 OR  s0> 1
    THEN PRINT "liegt nicht zwischen q1 und q2"
```

5.3 Schraffieren von Flächen

Bei der Herstellung von technischen Zeichnungen (Schraffieren von Schnittflächen) oder in der allgemeinen Datenverarbeitung zur besseren Kennzeichnung von Teilflächen wird ein Algorithmus zum automatischen Schraffieren von Flächen gebraucht. Mit den in Kapitel 5.2 besprochenen Gleichungen ist die Realisierung eines derartigen Programms möglich.

Die Gerade g der Abbildung 8 soll einer Schraffurgeraden entsprechen. Bekannt sind der Punkt P0(x0,y0) auf der Geraden g und der Richtungswinkel w, den die Schraffurgerade mit der x-Richtung bildet. Wird die unabhängige Variable der Schraffurgeraden g mit t bezeichnet, so ist g durch

```
x = x0 + t * cos(w)
y = y0 + t * sin(w)
```

gegeben. Durchläuft t die Werte von minus unendlich bis plus unendlich, so entsprechen den berechneten x,y-Werten in der x,y-Ebene Punkte, die auf einer Geraden durch P0 (x0,y0) mit dem Steigungswinkel w liegen. Für den Schnittpunkt Q der Geraden g und der Geraden g1 ergibt sich dann der spezielle t-Wert

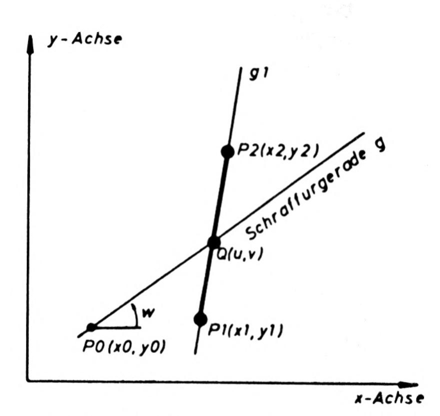


Abb. 8: Schraffurgerade g

$$t = \frac{(y0-y1)*\cos(w)-(x0-x1)*\sin(w)}{(y2-y1)*\cos(w)-(x2-x1)*\sin(w)}$$

In dem Programm

```
130 REM--- Schraffieren eines ge-    ---
140 REM--- schlossenen Polygonzuges ---
150 DATA 8, 5, 45
160 DATA -35, 35, -35,-35,  35,-35,  35, 20
170 DATA  20,-20,  10,20,    0,-20, -10, 35
190 DIM x(30), y(30), u(8), v(8)
195 :
200 i8 =  0.5: i9 =639.5: j8 =  0.5: j9 =399.5
210 x8 =-50  : x9 = 50  : y8 =-50  : y9 = 50
220 mx = (i9 - i8)/(x9 - x8):
    my = (j9 - j8)/(y9 - y8)
230 :
240 MODE 2:BORDER 3:INK 0,0:INK 1,26
260 READ n, d1, w
270 t1 = 1E+30: t2 = -t1: h = w*PI/180+0.00001
300 c1 = SIN(h): c2 = COS(h)
320 :
340 :  FOR i=1 TO n
350 :  READ x(i), y(i)
360 :  h = c2*y(i) - c1*x(i)
```

```
370 :   IF h<t1 THEN t1 = h: i1 = i
380 :   IF h>t2 THEN t2 = h: i2 = i
390 :   i2% = i8 + mx*(x(i)-x8)
400 :   j2% = j8 + my*(y(i)-y8)
410 :   IF i=1 THEN i3% = i2%: j3% = j2%:GOTO 430
420 :   MOVE i1%,j1%: DRAW i2%,j2%, 1
430 :   i1% = i2%: j1% = j2%
440 :   NEXT i
450 MOVE i2%,j2%: DRAW i3%,j3%, 1
460 x(n+1) = x(1): y(n+1) = y(1)
480 d2 = c2*(y(i2) - y(i1)) - c1*(x(i2) - x(i1))
490 :
500 FOR d=d1 TO d2-d1/2 STEP d1
520 x0 = x(i1) - c1*d
540 y0 = y(i1) + c2*d
560 m = 0: t1 = 1E+30: t2 = -t1
570 :
580 :   FOR i=1 TO n: i2 = i + 1
600 :   h = c2*(y(i2) - y(i)) - c1*(x(i2) - x(i))
620 :   IF h=0 THEN 800
640 :   t = (c2*(y0 - y(i)) - c1*(x0 - x(i)))/h
660 :   IF t<0 OR t>=1 THEN 800
680 :   m = m + 1
700 :   u(m) = x(i) + t*(x(i2) - x(i))
720 :   v(m) = y(i) + t*(y(i2) - y(i))
740 :   t = c2*(u(m) - u0) + c1*(v(m) - v0)
760 :   IF t<t1 THEN t1 = t: j1 = m
780 :   IF t>t2 THEN t2 = t: j2 = m
800 :   NEXT i
820 IF m=0 THEN
    PRINT "kein Schraff-Schnittpunkt vorhanden"
860 i2 = 1: IF ABS(j1-j2)=1
    THEN i2=2: u(m+1)=u(1): v(m+1)=v(1)
880 :   FOR i=i2 TO m STEP 2
890 :   i1% = i8 + mx*(u(i)-x8)
900 :   j1% = j8 + my*(v(i)-y8)
910 :   i2% = i8 + mx*(u(i+1)-x8)
920 :   j2% = j8 + my*(v(i+1)-y8)
930 :   MOVE i1%,j1%: DRAW i2%,j2%, 1
940 :   NEXT i
950 NEXT d
960 :
980 a$=INKEY$ : IF a$="" THEN 980
```

wird für jede Begrenzungsgerade des geschlossenen n-Ecks dieser t-Wert in der Zeile 640 berechnet. Die Koordinaten des Schnittpunktes Q(u,v) sind dann (siehe Zeilen 700,720):

```
u = x1 + t * (x2 - x1)
v = y1 + t * (y2 - y1)
```

Sind x0,y0,x1,y1,x2,y2 und w vorgegeben, so kann durch die Berechnung von t entschieden werden, ob der Schnittpunkt Q zwischen P1 und P2 liegt. Die Koordinaten u,v des Schnittpunktes können dann durch die obigen Gleichungen berechnet werden (s. Zeilen 700,720).

In dem angegebenen Programm sind die Eckpunkte P1,P2,...,Pn der Fläche vorgegeben, d. h. x1,y1,x2,y2,...,xn,yn sind bekannt. Die Koordinaten xi, yi sind in den Programmzeilen 160,170 abgelegt. Die Numerierung i=1,2,3,..., n der Eckpunkte wurde fortlaufend gewählt. Die zu schraffierende Fläche liegt immer links von dem Umlaufweg. Der Folgepunkt für Pn ist P1, d. h., es gilt x(n+1)=x1,y(n+1)=y1 (Zeile 460). In der Zeile 260 wird zunächst die Anzahl der Eckpunkte (n=8), der Abstand (d1=5) der Schraffurlinien und der Schraffursteigungswinkel (w=45°) eingelesen. Jene Eckpunkte, die den kleinsten bzw. größten senkrechten Abstand h von der Schraffurgeraden haben, werden in den Zeilen 360,370 und 380 ermittelt.

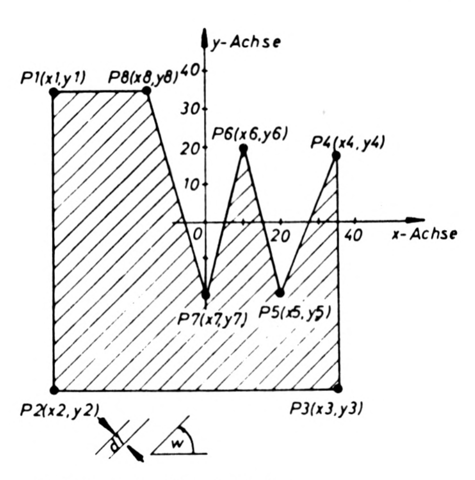
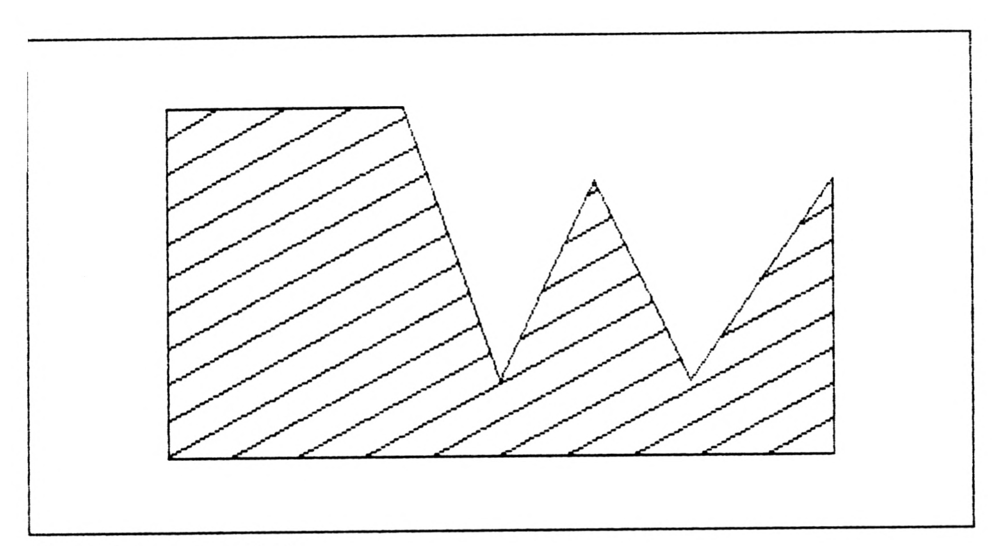


Abb. 9: Schraffierte Fläche

In der Schleife 340 bis 400 werden die Koordinaten xi,yi der n-Eckpunkte gelesen (Zeile 350) und die Nummer i1 bzw. i2 des Eckpunktes mit dem kleinsten bzw. größten Abstand von der Schraffurgeraden ermittelt. Die Linie wird mit Hilfe der Zeilen 390 bis 430 gezeichnet. Durch 450 wird der n-te Punkt mit dem 1. Punkt verbunden.

Der Wert d2 in der Zeile 480 entspricht dem Abstand zwischen P(i1) und P(i2). Die Schleife in Zeile 500 teilt diesen Abstand d2 in entsprechend viele Schraffurabstände d1 auf. Der Kern des Schraffurprogrammes wird durch die Zeilen 580 bis 800 gebildet. In der Zeile 600 wird der Nenner h der t-Gleichung ermittelt. Ist die Strecke P(i+1)−P(i) parallel zur Schraffurgeraden, so ist h=0. In diesem Fall werden die restlichen Anweisungen der Schleife (640−800) übersprungen. Die Zeile 640 berechnet den t-Wert. Liegt der Schnittpunkt (Schraffurgerade mit der Geraden durch P(i),Pi+1)) zwischen P(i) und P(i+1), so werden die Zeilen 680 bis 780 abgearbeitet. Die Berechnung der Koordinaten u(m),v(m) für den Punkt Q erfolgt in den Zeilen 700 und 720. Die aktuellen Schraffurgeraden werden durch die Zeile 930 gezeichnet. Dieses Verfahren wird jetzt mit dem Abstand d1 für die Ermittlung der weiteren Schraffurgeraden wiederholt.



Grafik 10: Schraffierte Fläche

6. Der Kreis

Die folgende Einführung ist für Leser gedacht, die bisher noch keinen Kontakt mit trigonometrischen Funktionen (sin(x),cos(x) usw.) hatten. Diese Funktionen werden in vielfacher Weise in der Computer-Grafik angewendet und sind im Standard-BASIC als Funktionen verfügbar. In der Abb. 10 ist ein Kreis dargestellt, der den Radius 1 hat. Einen solchen Kreis nennen wir Einheitskreis. Der Kreismittelpunkt und der Koordinatenursprung fallen zusammen. Die gestrichelt gezeichneten Strecken der Abb. 10 bezeichnen wir mit cos(t) und sin(t). Wird der Punkt Q auf dem Einheitskreis weitergeschoben, so ändern sich die Strecken cos(t),sin(t). Die Koordinaten von Q hängen davon ab, wie groß der Winkel t ist. Zum Messen des Winkels wollen wir die Länge t des Einheitskreis-Bogens verwenden. Z. B. entspricht dem Bogen t=pi/2 der Winkel t=90°. Einem Vollwinkel von t=360° entspricht am Einheitskreis die Bogenlänge t=2∗π. Ist ein Winkel in Grad vorgegeben, so kann dieser Winkel durch Multiplikation mit π/180=0.01745329 in das Bogenmaß umgerechnet werden. Ist der Winkel im Bogenmaß bekannt, so kann der Computer die Werte von sin(t) und cos(t) ausrechnen.

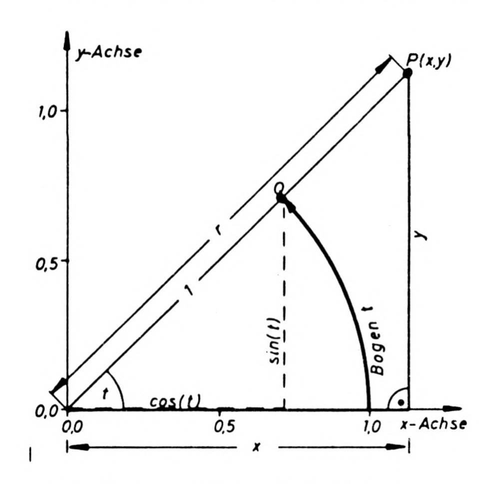


Abb. 10: Trigonometrische Funktionen am Einheitskreis

Geben wir am Computer

```
PRINT COS(1), SIN(1)
```

ein, so zeigt der Rechner die Werte 0.540302306, 0.841470985 an, die in der Abb. 10 den gestrichelten Linien entsprechen. Der Winkel 1 (im Bogenmaß) entspricht 57.2957795°. Soll der Computer die Werte von trigonometrischen Funktionen ausrechnen, so muß der Winkel immer im Bogenmaß vorliegen. Bei einem Funktionsaufruf wie z. B. sin(t) muß t im Bogenmaß verwendet werden. Liegt der Winkel in Grad vor, so kann mit Hilfe der Schneider-BASIC Anweisung

```
DEG
```

auf die Umrechnung in das Bogenmaß verzichtet werden, denn nach dieser Anweisung können die Winkel im Winkelgradmaß vorliegen. Der Rechner interpretiert den vorliegenden Winkel im Winkelgradmaß und nicht im Bogenmaß. Diese Interpretation wird durch

```
RAD
```

wieder aufgehoben. Danach muß der Winkel im Bogenmaß vorliegen.

Wenden wir den Strahlensatz auf Abb. 10 an, so ergeben sich die beiden Gleichungen

```
x = r * cos(t)
y = r * sin(t)
```

Durchläuft t die Werte von 0 bis 2π (im Bogenmaß) bzw. von 0° bis 360° (im Winkelgradmaß), je nach Interpretation, so ergeben diese beiden Gleichungen Punkte P(x,y) auf dem Kreis mit dem Radius r. Diese beiden Gleichungen stellen die Kreisgleichung in Parameterdarstellung dar, wobei der Kreismittelpunkt im Ursprung liegt. Für einen Kreis, dessen Mittelpunkt nicht im Koordinatenursprung (0,0) sondern im Punkt P0(x0,y0) liegt, lautet die Parameterdarstellung des Kreises

```
x = x0 + r * cos(t)
y = y0 + r * sin(t)
```

Durchläuft t die Werte von 0 bis 2π (im Bogenmaß) bzw. von 0° bis 360° (im Winkelgradmaß), so liegen die Punkte P(x,y) auf einem Kreis mit dem Radius r und dem Kreismittelpunkt P0(x0,y0).

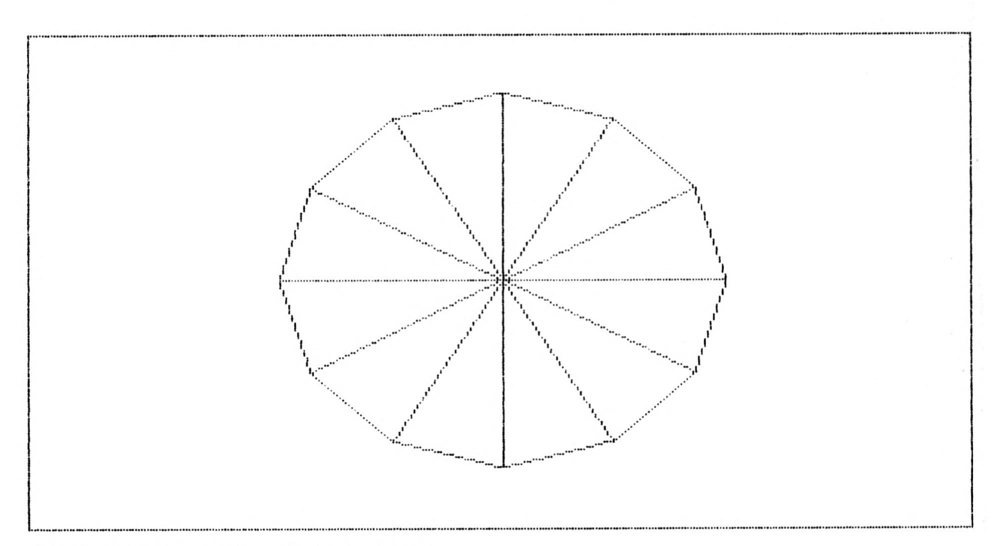
Das folgende Programm

```
100 REM--- Wertetabelle fuer Kreispunkte ---
110 r = 5: x0 = -1: y0 = 2
120 :
130 PRINT "     t          x          y"
140 PRINT "   (Grad) "
150 PRINT "   ---------------------------"
160 :
170 DEG
180 FOR t=0 TO 360 STEP 30
190 :   x = x0 + r*COS(t)
200 :   y = y0 + r*SIN(t)
210 :   PRINT USING "######.#";    t;
220 :   PRINT USING "#######.##"; x;
230 :   PRINT USING "#######.##"; y;
240 :   PRINT
250 NEXT t
```

berechnet eine Wertetabelle für die Punkte P(x,y) eines Kreises mit dem Radius 5 und dem Mittelpunkt P0(−1,2). Die Umrechnung des Winkels t (Grad) in das Bogenmaß kann entfallen, da in der Zeile 170 die Interpretation der Argumente von trigonometrischen Funktionen auf das Winkelgradmaß eingestellt wird.

Für die formatierte Ausgabe wird der PRINT USING-Befehl verwendet. Die erste Zeichenkette hinter PRINT USING gibt die Form an, in der die Werte ausgegeben werden sollen. Die zweite Zeichenkette hinter PRINT USING entspricht den auszugebenden Variablen. Das PRINT in der Zeile 240 beendet eine Ausgabezeile. Mit dem PRINT USING-Befehl kann in einfacher Weise eine tabellenförmige Ausgabe erzeugt werden, bei der die Dezimalpunkte der Zahlenkolonnen exakt untereinander stehen.

```
   t           x          y
 (Grad)
--------------------------------
   0.0       4.00       2.00
  30.0       3.33       4.50
  60.0       1.50       6.33
  90.0      -1.00       7.00
 120.0      -3.50       6.33
 150.0      -5.33       4.50
 180.0      -6.00       2.00
 210.0      -5.33      -0.50
 240.0      -3.50      -2.33
 270.0      -1.00      -3.00
 300.0       1.50      -2.33
 330.0       3.33      -0.50
 360.0       4.00       2.00
```



Grafik 11: Parameterdarstellung des Kreises

Zum Zeichnen eines Kreises ist das folgende Programm nur geringfügig gegenüber dem vorhergehenden geändert worden:

```
300 REM--- Kreis zeichnen mit der ---
310 REM--- Parameterdarstellung   ---
320 MODE 1:BORDER 3:INK 0,12:INK 1,26
330 r = 150: x0 = 320: y0 = 200
340 :
```

```
350 ORIGIN x0, y0
360 PLOT 0, 0, 1
370 DEG
380 FOR t=0 TO 360 STEP 30
390 :  x = r*COS(t): y = r*SIN(t)
400 :  IF t>0 THEN MOVE x1, y1: DRAW x, y, 1
410 REM--- IF t>0 THEN MOVE 0, 0: DRAW x, y, 1
420 :  x1 = x: y1 = y
430 NEXT t
440 RAD
450 :
460 a$=INKEY$: IF a$="" THEN 460
```

In der Zeile 320 wird zunächst der normalauflösende Bildschirm mit der Zeichenfarbe weiß (26) auf gelbem (12) Hintergrund eingeschaltet. Danach wird der Ursprung des Bildschirmkoordinatensystems in den Mittelpunkt P0(x0,y0) des Kreises verlegt (Zeile 350). Der Mittelpunkt P0(x0,y0) hat jetzt die Koordinaten (0,0) und wird mit Hilfe der Anweisung

```
PLOT 0, 0, 1
```

markiert. Mit der Schrittweite von 30° (Zeile 380) werden aufeinanderfolgende Kreispunkte P(x,y) berechnet (Zeile 390). Bei dieser Berechnung entfällt das Umrechnen von t in das Bogenmaß, da der Rechner durch Zeile 370 auf Gradmaßinterpretation umgeschaltet wird. Darüber hinaus ist die Addition der Kreismittelpunktskoordinaten x0 und y0 zu den Koordinaten der Kreispunkte P(x,y) überflüssig, weil der Ursprung des Bildschirmkoordinatensystems im Mittelpunkt P0(x0,y0) liegt. Die aufeinanderfolgenden Kreispunkte werden durch eine Linie (Zeile 400) miteinander verbunden. Bei einer Schrittweite von 30° entsteht ein 12-Eck. Wird die Schrittweite verkleinert (z. B. 1°), so entsteht ein Kreis. Wenn wir die Zeile 410 aktivieren, indem wir „REM---“ löschen, so wird jeder berechnete Kreispunkt P(x,y) durch eine Linie mit dem Kreismittelpunkt P0(x0,y0) verbunden, und wir erhalten Grafik 11.

7. Die Ellipse

Eine Ellipse entspricht einem gestauchten Kreis. Daher können viele Bemerkungen aus dem vorhergehenden Kapitel über Kreise übernommen werden. In der Computergrafik wird als Ellipsengleichung die Parameterdarstellung bevorzugt verwendet.

Diese Darstellung

```
x = x0 + a * cos(t)
y = y0 + b * sin(t)
```

benutzt den Parameter t zur Berechnung der x,y-Koordinaten der Ellipsenpunkte P(x,y). Der vorgegebene Ellipsenmittelpunkt hat die Koordinaten x0,y0. Die vorgegebenen Ellipsenhalbachsen sind a und b. Der Sonderfall a=b ergibt einen Kreis mit dem Mittelpunkt P0(x0,y0) und dem Radius r=a=b. Die Brennpunkte F1,F2 der Ellipse sind durch den Abstand

```
e = sqr (a*a - b*b)
```

vom Ellipsenmittelpunkt festgelegt.

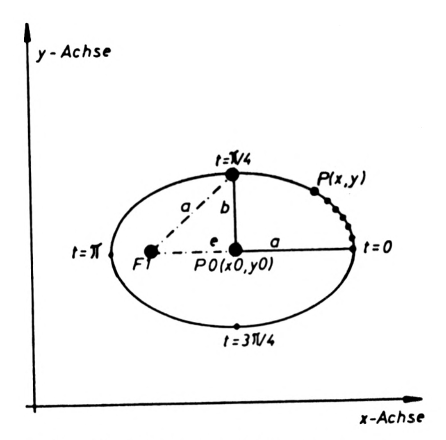


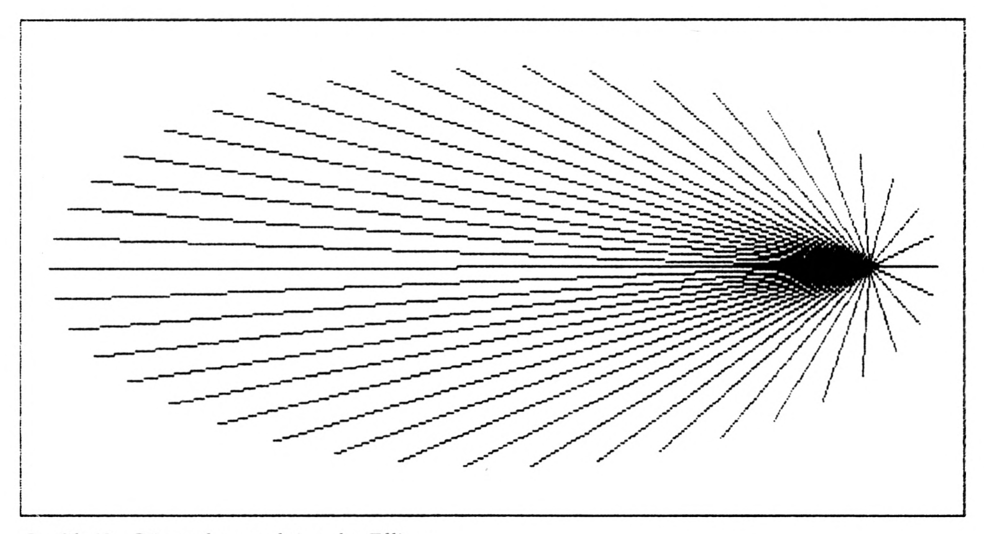
Abb. 11: Ellipsenbezeichnungen

In dem folgenden Programm wird entsprechend der Gärtner-Konstruktion von einem Ellipsenpunkt P(x,y) zum Brennpunkt F2 (Zeile 465) und gegebenenfalls von P zu F1 (Zeile 460) eine Linie gezeichnet (Grafik 12). In den Zeilen 470 und 475 wird ein neuer Ellipsenpunkt P(x,y) berechnet und in Zeile 485 wird gegebenenfalls von P1 nach P die Ellipsen-Rand-Linie gezeichnet.

Werden zum Beispiel die Halbachsen a,b gemäß Zeile 480 gekürzt, so erhalten wir (t=0 bis t=15) eine modifizierte Grafik. In ähnlicher Weise können auch andere Ellipsenparameter verändert werden. Durch einen Programmlauf wird die Wirkung der Programmänderungen sichtbar (Grafik 12).

```
400 REM --- Ellipsen zeichnen ---
410 MODE 2:BORDER 2:INK 0,1:INK 1,24
415 x0 = 320: y0 = 200
420 a  = 300: b  = 160
430 e  = SQR(a*a - b*b)
435 x1 = x0 + a: y1 = y0
440 :
450 FOR t=0 TO 6.4 STEP 0.15
```

```
455 :    REM--- MOVE x1, y1: DRAW x0, y0, 1
460 :    REM--- MOVE x1, y1: DRAW x0-e, y0, 1
465 :   MOVE x1, y1: DRAW x0+e, y0, 1
470 :   x = x0 + a * COS(t)
475 :   y = y0 + b * SIN(t)
480 :    REM--- a=a/1.02 : b=b/1.01
485 :    REM--- MOVE x1, y1: DRAW x, y, 1
490 :   x1 = x: y1 = y
495 NEXT t
500 :
505 a$=INKEY$ : IF a$="" THEN 505
```



Grafik 12: Gärtnerkonstruktion der Ellipse

8. Die Parabel

Eine nach oben bzw. nach unten offene Parabel in einem x,y-System ist durch die Gleichung

```
y(x) = a + b*x + c*x*x
```

gegeben. Wenn wir eine solche Parabel zeichnen wollen, so müssen die Koeffizienten a,b,c bekannt sein. Auf das automatische Zeichnen von solchen Funktionen werden wir später eingehen. An dieser Stelle wollen wir eine Parabel darstellen, die durch die vorgegebenen Punkte P0(x0,y0), P1(x1,y1), P2(x2,y2) geht. Die gesuchte Parabelgleichung kann leicht aus dem Interpolationspolynom nach NEWTON erhalten werden. Hierzu berechnet man aus den vorgegebenen Koordinaten x0,y0,x1,y1,x2,y2 die Werte

$$d0 = \frac{y1-y0}{x1-x0}$$

$$d1 = \frac{y2-y1}{x2-x1}$$

$$d2 = \frac{d1-d0}{x2-x0}$$

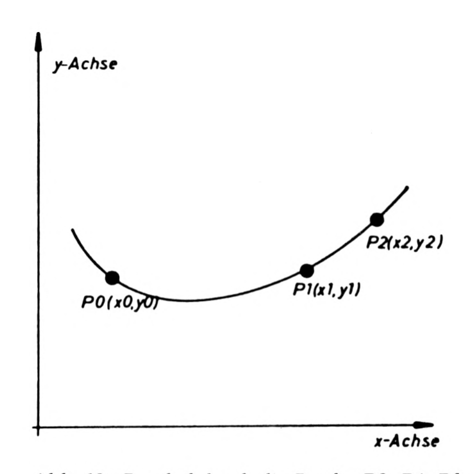


Abb. 12: Parabel durch die Punkte P0, P1, P2

und erhält damit die Gleichung

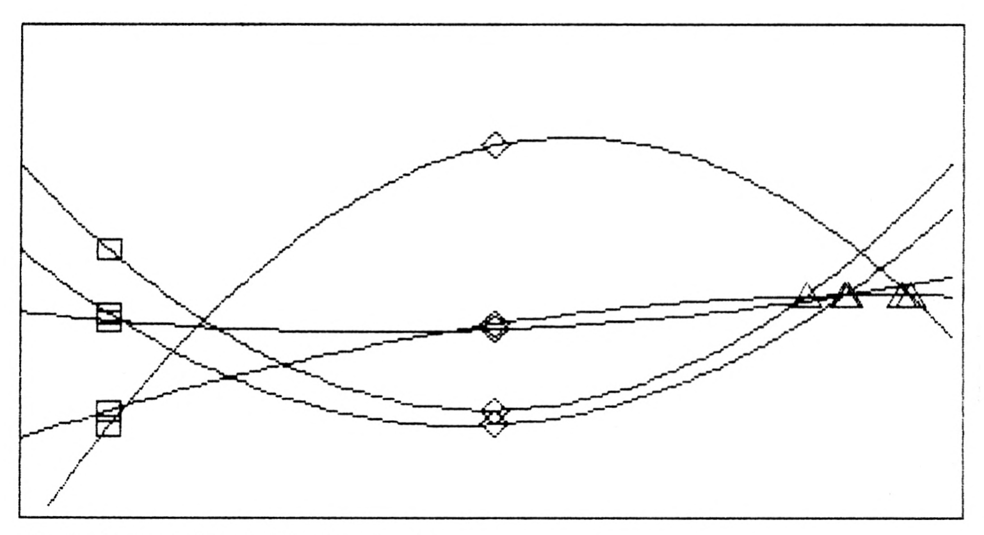
```
y(x) = y1 + (x - x1) * (d0 + d2*(x-x0))
```

der Parabel, die durch die Punkte P0,P1,P2 geht. Diese Formel können wir dadurch überprüfen, indem wir x=x1 einsetzen. Wir erhalten y=y1. Für x=x0 ergibt sich y=y0 und für x=x2 ergibt sich y=y2. In dem Programm

```
100 REM--- Parabel durch 3 Punkte ---
110 REM--- zeichnen nach Newton   ---
120 MODE 2:BORDER 3:INK 0,3:INK 1,26
130 MOVE 0, 0: DRAW 639, 0, 1: DRAW 639,399, 1
140 DRAW 0,399, 1: DRAW 0, 0, 1
150 :
160 FOR k=1 TO 5
170 y0 = 20 + 319*RND(1): x0 =  60
180 y1 = 20 + 359*RND(1): x1 = 320
190 x2 = 639 - y0/2      : y2 = 180
```

```
200 :
210 MOVE x0-8, y0-8: DRAW x0-8, y0+8, 1
220 DRAW x0+8, y0+8, 1: DRAW x0+8, y0-8, 1
225 DRAW x0-8, y0-8, 1
230 :
240 MOVE x1-10, y1: DRAW x1, y1+10, 1
245 DRAW x1+10, y1, 1: DRAW x1, y1-10, 1
250 DRAW x1-10, y1, 1
260 :
270 MOVE x2, y2+10: DRAW x2+10, y2-10, 1
280 DRAW x2-10, y2-10, 1: DRAW x2, y2+10, 1
290 :
300 d0 = (y1-y0)/(x1-x0)
310 d1 = (y2-y1)/(x2-x1)
320 d2 = (d1-d0)/(x2-x0)
330 :
340 DEF FN y(x) = y1 + (x-x1) * (d0 + d2*(x-x0) )
350 :
360 xa = 0: xb = 639: dx = 10: ya = FN y(xa)
370 :
380 :  FOR x=xa+dx TO xb STEP dx: y = FN y(x)
390 :   IF y<0 OR y>399 OR ya<0 OR ya>399 THEN 410
400 :   MOVE xa, ya: DRAW x, y, 1
410 :   xa = x: ya = y
420 :  NEXT x
425 :
430 NEXT k
435 :
440 a$=INKEY$: IF a$="" THEN 440
```

wird in Zeile 120 der hochauflösende Bildschirm eingeschaltet. Als Zeichenfarbe wird weiß (26) und als Hintergrund- und Rahmenfarbe rot (3) verwendet. In den Zeilen 170,180,190 werden die Koordinaten der Punkte P0,P1,P2 gewählt. Diese Punkte werden in den Zeilen 210 bis 225,240 bis 250 und 270,280 markiert. In den Zeilen 300,310,320 werden die Hilfsgrößen d0,d1,d2 für die Parabelgleichung berechnet. Diese Parabel wird durch die Zeile 340 definiert. Die Schleife 380 bewirkt, daß x von xa bis xb läuft. In diesem Bereich werden die y-Werte berechnet und aufeinanderfolgende Punkte durch Linien verbunden (Zeile 400). Die Zeile 390 überspringt das Zeichnen, wenn Parabelstücke außerhalb des Bildschirmes liegen. Durch die k-Schleife (Zeile 160) werden 5 Parabeln gezeichnet (s. Grafik 13).



Grafik 13: Parabel durch 3 gegebene Punkte

9. Demo-Grafiken

In diesem Kapitel über Demo-Grafiken werden kurze Programme dargestellt, die geeignet erscheinen, ansprechende visuelle Darstellungen zu erzeugen. Der mathematische Hintergrund dieser Programme wird erläutert, um so dem Leser die Gelegenheit zum weiteren Experimentieren zu geben. In vielen Fällen ist es leicht möglich, durch geringe Programmänderungen künstlerische Assoziationen hervorzurufen.

9.1 Diagonalnetz

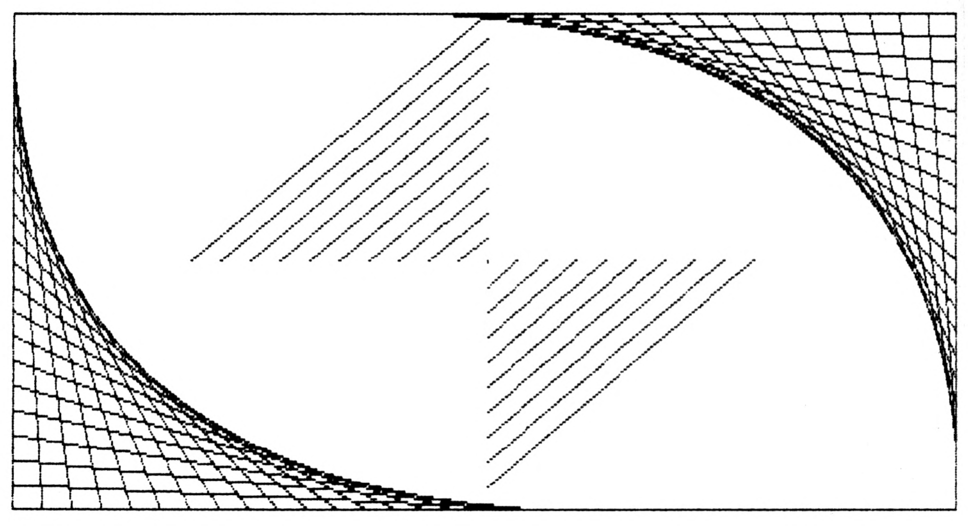
Einfache und interessante Grafiken lassen sich durch ein systematisch angeordnetes Liniennetz erzeugen. Dabei werden von einem Punkt ausgehende Linien zu verschiedenen anderen Punkten gezogen. Durch den Wechsel des Ausgangspunktes entstehen ansprechende Computergrafiken. Zum Beispiel ergibt sich ein gewölbtes Netz (s. Grafik 14), wenn von den Punkten P1(0,399–y) zu den Punkten P2(y,0) Linien für y=0 bis 399 gezogen werden. Ein derartiges Programm sieht in Schneider-BASIC dann etwa so aus:

```
20 REM--- Netz zeichnen ---
30 MODE 2:BORDER 3:INK 0,3:INK 1,26
35 :
40 FOR y=0 TO 399 STEP 20
50 :   MOVE   0,399-y: DRAW     y,  0, 1
60 :   MOVE 639,     y: DRAW 639-y,399, 1
70 :   MOVE 320,399-y: DRAW 120+y,200, 1
80 NEXT y
85 :
90 a$=INKEY$: IF a$="" THEN 90
```

Die Anweisungen in Zeile 50 verfahren nach dem obigen Schema. Die Anweisungen in der Zeile 60 sorgen für eine Spiegelung des Bildes auf die rechte Bildschirmseite. Die Zeile 70 erzeugt die Figur in der Bildmitte.

Das nun folgende Programm arbeitet mit zwei ineinander geschachtelten Schleifen. Beide laufen von 0 bis 640, mit der Schrittweite 80 (Zeilen 230,240). Nachdem in Zeile 210 die hochauflösende Grafik eingeschaltet wurde, werden weiße (26) Linien auf rotem (3) Grund von P1(x1,y1) nach P2(x2,y2) gezogen. Die x-Koordinate x2 von P2 wird nach jeder Linie um 80 Bildpunkte verschoben (Schrittweite), während die y-Koordinate y2 den Wert 0 beibehält (Zeile 240). Erreicht x2 den Wert 640, so wird der Punkt P1, gemäß der Schleife in Zeile 230, um 80 Bildpunkte (Schrittweite) nach rechts verschoben. Mit dem neuen Anfangspunkt erfolgen die gleichen Ausführungen.

```
200 REM--- Diagonalnetz ---
210 MODE 2:BORDER 3:INK 0,3:INK 1,26
220 :
230 FOR x1=0 TO 640 STEP 80
235 y1 = 400
237 REM--- y1 = 400-(x1-320)^2/400
240 :   FOR x2=0 TO 640 STEP 80
250 :   MOVE x1, y1: DRAW x2, y2, 1
260 :   NEXT x2
270 NEXT x1
290 :
295 a$=INKEY$: IF a$="" THEN 295
```

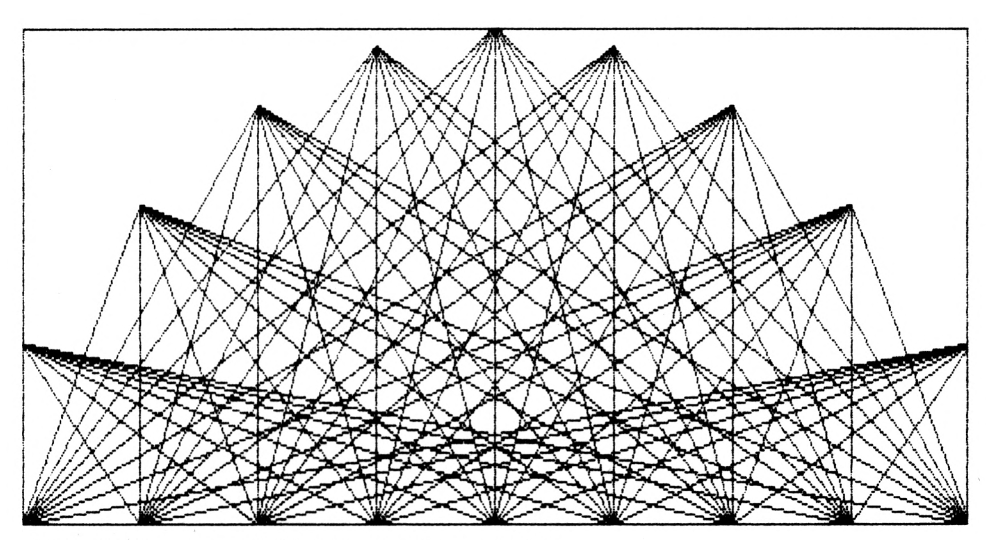


Grafik 14: Liniennetz

Wird in Zeile 237 „REM---" entfernt, so wird y1 in Abhängigkeit von x1 berechnet und die y-Werte parabelförmig aufgetragen, so daß die Grafik 15 entsteht.

9.2 Moiree-Effekt

Das Wort „Moiree" kommt von „Moare" und bedeutet soviel wie Wasserlinienmusterung. Z. B. treten solche Muster bei der Überlagerung von Streifengittern auf dem Bildschirm auf. Dies ähnelt der physikalischen Interferenz in Wellenfronten. An dieser Stelle soll auf den Artikel von G. Oster und N. Nishijima mit dem Titel „Moiree Patterns" in Scientific American 1963, Seite 54 hingewiesen werden.



Grafik 15: Netz mit parabelförmig angeordneten Punkten

Bei dem folgenden Programm benutzen wir die Möglichkeit, Bildschirmpunkte zu invertieren; d. h., einen gesetzten Punkt zu löschen und umgekehrt. Dies geschieht durch eine Änderung des Farbstiftmodus. Im Normalfall werden beim Zeichnen von Grafiken vorher gezeichnete Grafiken einfach überdeckt. Ein Punkt, der mehrmals in verschiedenen Farben gezeichnet wurde, wird also in der Farbe des zuletzt benutzten Farbstiftes dargestellt. Demnach findet eine Überdeckung mehrfach gesetzter Bildschirmpunkte statt. Um dies zu ändern, muß an den Rechner eine bestimmte Kommando-Sequenz gesendet werden. Dieses Kommando (Zeile 320) lautet:

```
PRINT CHR$(23); CHR$(1);
```

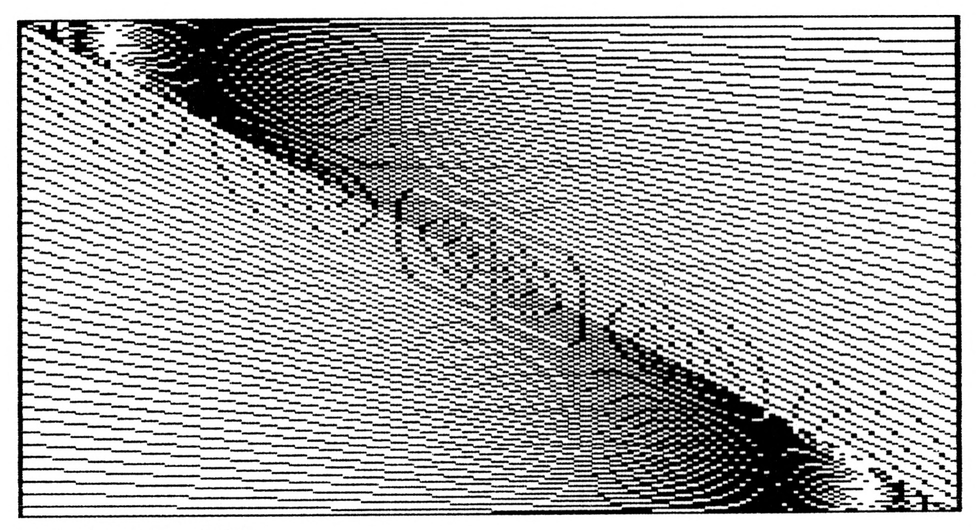
Von jetzt an werden gesetzte Punkte nicht mehr einfach überdeckt. Um diesen Grafikfarbstiftmodus wieder aufzuheben, muß folgendes Kommando gegeben werden (Zeile 380):

```
PRINT CHR$(23); CHR$(0);
```

Da wir eine mehrfarbige Grafik erstellen wollen, arbeitet das Programm im Vielfarbmodus (MODE 0). Dieser Modus wird in der Zeile 305 eingeschaltet. Die gegenüber dem hochauflösenden Bildschirm deutlich geringere Auflösung beträgt 160x200 Punkte.

In der Zeile 310 werden den Farbstiften 1,2,3 und 4 die Farben schwarz (0), rot (3), helles blaugrün (20), und hellgelb (24) zugeordnet. Durch BORDER 10 und INK 0, 13 wird die Rahmenfarbe blaugrün (10) und die Hintergrundfarbe weiß (13) eingeschaltet. Vom linken oberen Bildschirm-Eckpunkt P(0,400) und vom unteren rechten Eckpunkt P(639,0) werden Linien im Abstand von 20 Bildpunkten (Schrittweite in Zeile 335) zum gegenüberliegenden y-Rand gezogen. Die linken Diagonalen werden mit dem Farbstift 1 also in schwarz, die rechten mit dem Farbstift 2 also in rot erstellt.

Die zweite Schleife (Zeile 360) veranlaßt das Linienzeichnen von P(0,400) und P(639,0) zum gegenüberliegenden x-Rand mit y-Werten von 0 bis 399 in der Schrittweite 10 (Zeile 360).



Grafik 16: Moiree-Effekt

Wird die Zeile 325 aktiviert (REM--- löschen), so lassen sich ellipsenförmige ineinandergeschachtelte Figuren erkennen, die im Bereich der Eckpunkte besonders deutlich sind. Diese Figuren entstehen durch die geringe Auflösung und die damit verbundenen Spalten- bzw. Zeilensprünge beim Zeichnen von Linien.

```
300 REM--- Moiree-Effekt ---
305 MODE 0: BORDER 10:INK 0,13
310 INK 1,0:INK 2,3:INK 3,20:INK 4,24
315 :
320 PRINT CHR$(23);CHR$(1);
325 REM--- GOTO 360
330 :
335 FOR x=0 TO 639 STEP 20
340 :  MOVE  0 ,400: DRAW x, 0 , 1
345 :  MOVE 639, 0 : DRAW x,400, 2
350 NEXT x
355 :
360 FOR y=0 TO 399 STEP 10
365 :  MOVE  0 ,400: DRAW 639,y, 3
370 :  MOVE 639, 0 : DRAW  0 ,y, 4
375 NEXT y
```

```
380 PRINT CHR$(23);CHR$(0);
385 :
390 a$=INKEY$: IF a$="" THEN 390
```

9.3 Geschachtelte Sechsecke

Mit Hilfe der Ellipsengleichung werden in dem folgenden Programm Sechsecke erzeugt. Die Parameterdarstellung (Zeilen 480,490) benutzt den Parameter t zur Berechnung der x,y-Koordinaten der Eckpunkte. Der Mittelpunkt (x0,y0), sowie die Ellipsen-Halbachsen a,b sind in der Zeile 430 festgelegt. Die Variable n wird in Zeile 415 mit der Anzahl der gewünschten Eckpunkte belegt. Für Sechsecke ist n=6. Der Zentriwinkel w0 wird in Zeile 470 durch die Multiplikation mit der jeweiligen Eckpunktnummer modifiziert und ergibt den Ellipsenparameter t. Dieser Parameter wird nicht in das Bogenmaß umgerechnet, da der Rechner nach Zeile 450 mit Gradmaß arbeitet.

Die Anzahl der ineinandergeschachtelten Sechsecke ist auf 15 begrenzt (Zeile 520). Jeder Eckpunkt wird mit dem nachfolgenden durch eine Linie verbunden (Zeile 550). Danach erhalten die aktuellen Feldvariablen x(),y() die Koordinatenwerte des nächsten (kleineren) Sechseckes (Zeile 560 und 570). Diese Werte entsprechen den Mittelpunkten der bereits gezeichneten Linien. Ein Schließen des Sechseckes bewirkt die Zeile 590, indem für den (n+1)-ten Punkt der erste Punkt gesetzt wird. Durch den Befehl MODE 1 in der Zeile 410 wird der normalauflösende Bildschirm eingeschaltet. Die Anweisung BORDER 12 in der gleichen Zeile bewirkt das Einschalten der Rahmenfarbe gelb (12).

Durch die Zeile 420 wird den vier Tintennummern 0,1,2 und 3 jeweils eine Farbnummer zugeordnet. Dabei wird z. B. der Tinte mit der Nummer 2 die Farbe hellgrün (18) zugewiesen. Auf dem Bildschirm erscheinen also alle Linien, die mit dieser Tinte gezeichnet werden in hellgrüner Farbe (18).

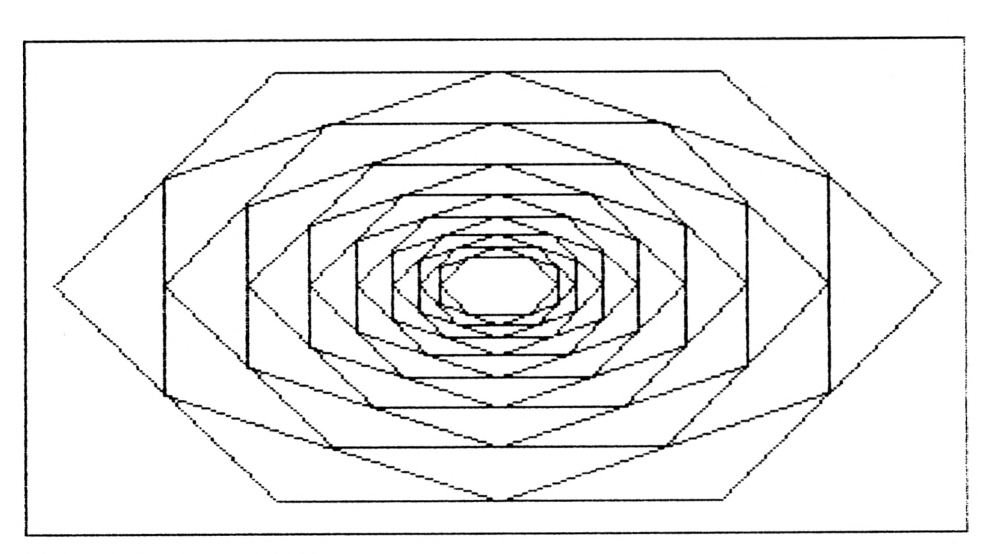
Durch die Programmzeilen

```
540 f = f+1
550 MOVE x(k),y(k): DRAW x(k+1),y(k+1), 1 + f MOD 3
```

wird für jede Linie periodisch eine neue Tinte (eine neue Farbe) aus den Tinten 1,2 und 3 ausgewählt. Dardurch wird also die 1. Linie hellgrün die 2. hellgelb und die dritte blau gezeichnet.

```
400 REM--- geschachtelte Sechsecke ---
410 MODE 1: BORDER 12
420 INK 0,1: INK 1,14: INK 2,18: INK 3,24
430 x0 = 320: y0 = 200: a = 300: b = 200
440 n = 6: w0 = 360/n
445 :
450 DEG
460 FOR k=0 TO n
```

```
470 :   t = k * w0
480 :   x(k) = x0 + a*COS(t)
490 :   y(k) = y0 + b*SIN(t)
500 NEXT k
510 RAD
515 :
520 FOR j=1 TO 15
530 :   FOR k=0 TO n-1
540 :   f = f+1
550 :   MOVE x(k), y(k):
        DRAW x(k+1),y(k+1), 1 + f MOD 3
560 :   x(k) = ( x(k) + x(k+1) )/2
570 :   y(k) = ( y(k) + y(k+1) )/2
580 :   NEXT k
590 :   x(n) = x(0): y(n) = y(0)
595 NEXT j
597 :
598 a$=INKEY$: IF a$="" THEN 598
```

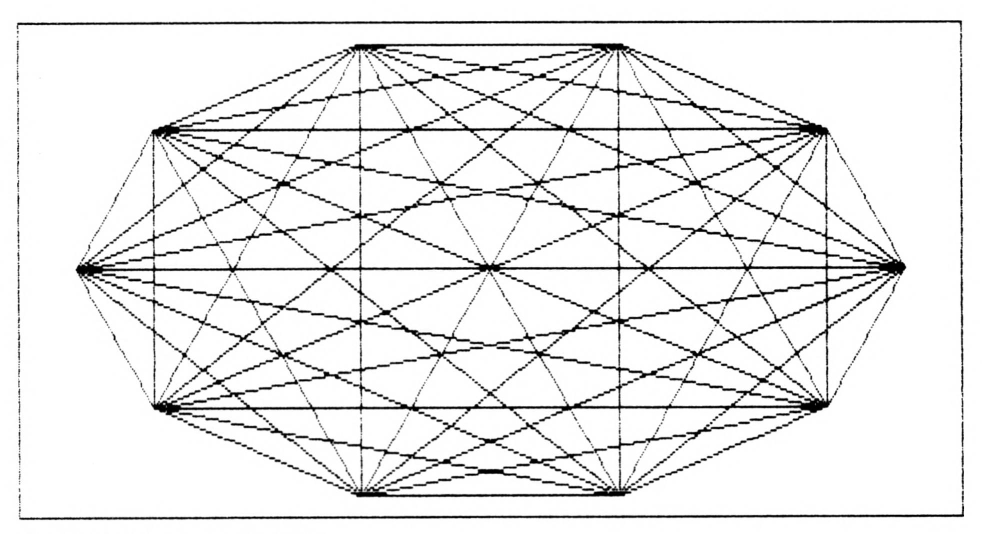


Grafik 17: Geschachtelte Sechsecke

9.4 Diagonalen im n-Eck

Auf den gleichen Grundlagen wie die Programme „Geschachtelte Sechsecke“ und „Eingeschriebene Vierecke“ basiert das Programm „Diagonalen im n-Eck“.

In den Zeilen 680 und 690 wird die bekannte Parameterdarstellung der Ellipsengleichung mit dem Ellipsenparameter t, dem Mittelpunkt (x0,y0), und den Halbachsen a,b verwendet. Der hochauflösende Bildschirm wird in der Zeile 610 mit der Hintergrundfarbe rot (3) und der Zeichenfarbe weiß (26) aktiviert. In der Schleife zwischen Zeile 660 und Zeile 700 werden die x- und y-Koordinaten der n-Eckpunkte berechnet. Die beiden ineinandergeschachtelten Schleifen der Zeilen 730 und 740 dienen dazu, von jedem errechneten Eckpunkt aus eine Linie (Zeile 750) zu jedem anderen Eckpunkt zu zeichnen. Um ein „Nachziehen“ bereits vorhandener Linien zu umgehen, läuft die zweite Schleife ab dem um 1 erhöhten aktuellen Wert der ersten Schleife, d. h. von (k+1) bis n. Ist z. B. eine Linie vom Eckpunkt 1 zum Eckpunkt 2 gezogen worden, so ist das Zeichnen der Linie vom Punkt 2 zum Punkt 1 nicht mehr erforderlich (Zeile 740).



Grafik 18: Diagonalen im n-Eck

Wird die Anzahl der Eckpunkte n (Zeile 630) erhöht (n>10), so ist eine zusätzliche Dimensionierung erforderlich. Es kann beispielsweise durch den Einschub der folgenden Zeile

```
640 DIM x(n), y(n)
```

dem Rechner mitgeteilt werden, daß n-Speicherplätze für die indizierten Variablen x(n) und y(n) benötigt werden.

```
600 REM--- Diagonalen im n-Eck ---
610 MODE 2:BORDER 2:INK 0,3:INK 1,26
620 x0 = 320: y0 = 200: a = 280: b = 190
630 n = 10: w0 = 360/n
640 :
650 DEG
660 FOR k=1 TO n
670 :  t = k * w0
680 :  x(k) = x0 + a*COS(t)
690 :  y(k) = y0 + b*SIN(t)
700 NEXT k
710 RAD
720 :
730 FOR k=1 TO n-1
740 :  FOR j=k+1 TO n
750 :  MOVE x(k), y(k): DRAW x(j), y(j), 1
760 :  NEXT j
770 NEXT k
780 :
790 a$=INKEY$: IF a$="" THEN 790
```

9.5 Röhrengrafik

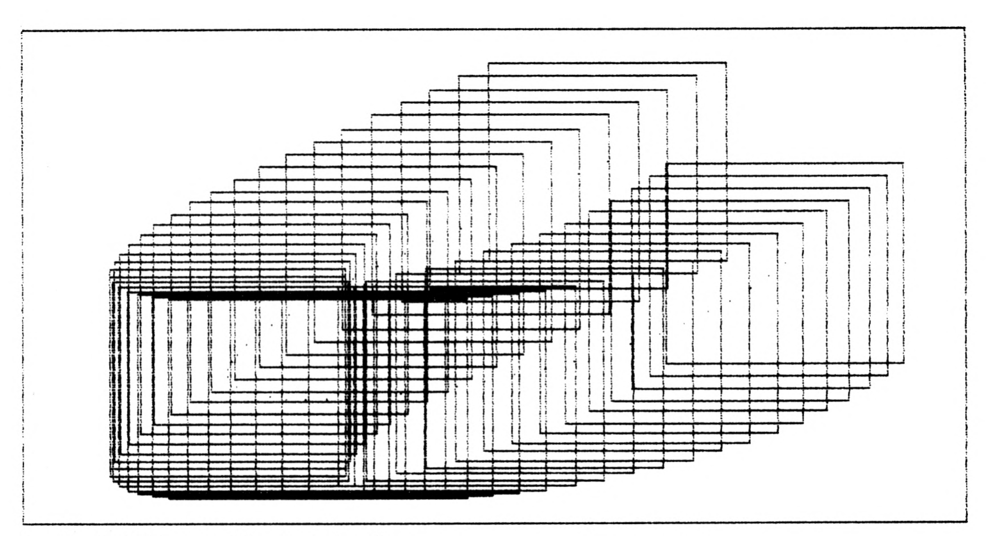
Bei den nächsten drei Demoprogrammen werden wir uns mit dem Zeichnen von Kreisen befassen. Um interessante Grafiken zu erzeugen, verfahren wir ebenso wie bei den vorhergehenden Programmen. Wir reihen systematisch eine Vielzahl von Kreisen aneinander, so daß Feder-, Spiralen-, Röhrenbilder usw. entstehen. Der Nachteil beim Zeichnen von Kreisen ist die langsame Verarbeitungsgeschwindigkeit. Daher ist es beim Ausprobieren unter Umständen sinnvoll, das Zeichnen von Kreisen durch das bedeutend schnellere Zeichnen von Rechtecken zu ersetzen. Zum Überprüfen können wir uns den Mittelpunkt durch

```
PLOT x, y, 1
```

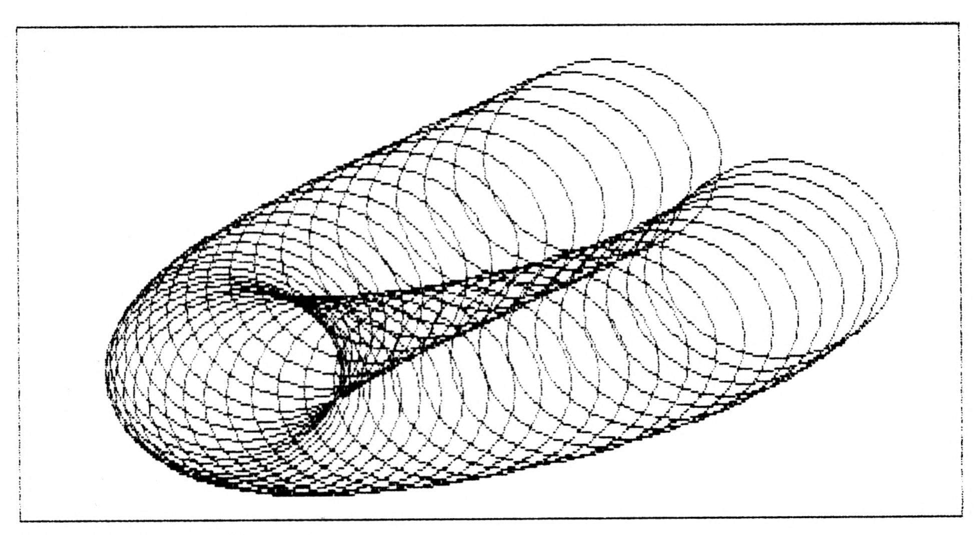
mit ausgeben lassen. Bei dem folgenden Programm werden zum Testen die Zahlen 170 bis 200 aktiviert (REM--- löschen) und die Zeilen 220 bis 290 deaktiviert (REM hinter Zeilennummer einschieben). Es entsteht die Grafik 19.

Werden dagegen die Zeilen 220 bis 290 wieder aktiviert (170—200 deaktivieren), so ergibt sich die Grafik 20.

Das Demoprogramm für die Röhrengrafik berechnet nach dem Einschalten des hochauflösenden Bildschirmes in den Zeilen 150,160 die Mittelpunktkoordinaten x,y. Dann wird durch die Schleife in der Zeile 240 ein Kreis (Grafik 20) mit dem festen Radius 80 um die-



Grafik 19: Röhrengrafik mit Rechtecken



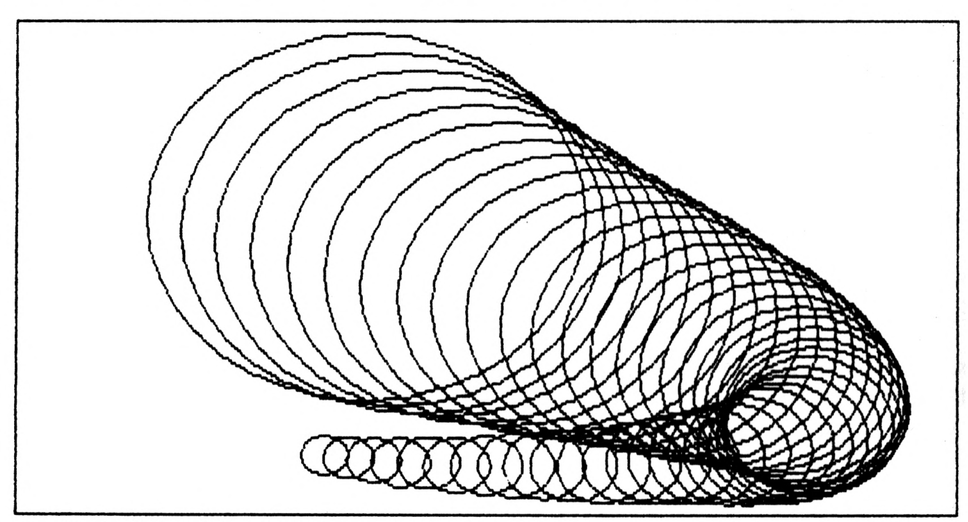
Grafik 20: Röhrengrafik mit Kreisen

sen Mittelpunkt gezeichnet. Die Anzahl der Kreise ist durch die Schleife in Zeile 140 festgelegt.

```
100 REM--- Roehren-Grafik 1 ---
110 MODE 2:BORDER 3:INK 0,1:INK 1,26
120 r = 80
130 :
```

```
140 FOR t=1 TO 10 STEP 0.2
150 :   x = 340 +200*COS(t/2)
160 :   y = 260 -160*SIN(t/3)
170 REM--- PLOT x, y, 1
180 REM--- MOVE x-r, y+r: DRAW x+r, y+r, 1
190 REM--- DRAW x+r, y-r, 1: DRAW x-r, y-r, 1
200 REM--- DRAW x-r, y+r, 1
210 :
220 : ORIGIN x, y
230 : DEG
240 : FOR a=0 TO 360 STEP 10
250 :   x2 = r*COS(a): y2 = r*SIN(a)
260 :   IF a>0 THEN MOVE x1, y1:DRAW x2, y2, 1
270 :   x1 = x2: y1 = y2
280 : NEXT a
290 : RAD
300 :
310 NEXT t
320 :
330 a$=INKEY$: IF a$="" THEN 330
```

Bei dem folgenden Programm werden in den zeilen 330,335 die Mittelpunkte der Kreise berechnet. Die Radien der Kreise ändern sich (Zeile 340). Der Radius wird jetzt in Abhängigkeit von t berechnet ($t^3/4$). Dies führt zu einem kontinuierlichen Anwachsen der Halbmesser, wobei t durch die Schleife (Zeile 320) vorgegeben wird. Anstelle von Kreisen (Zeilen 400–450) können wir auch Rechtecke (Zeilen 370–390) zeichnen.



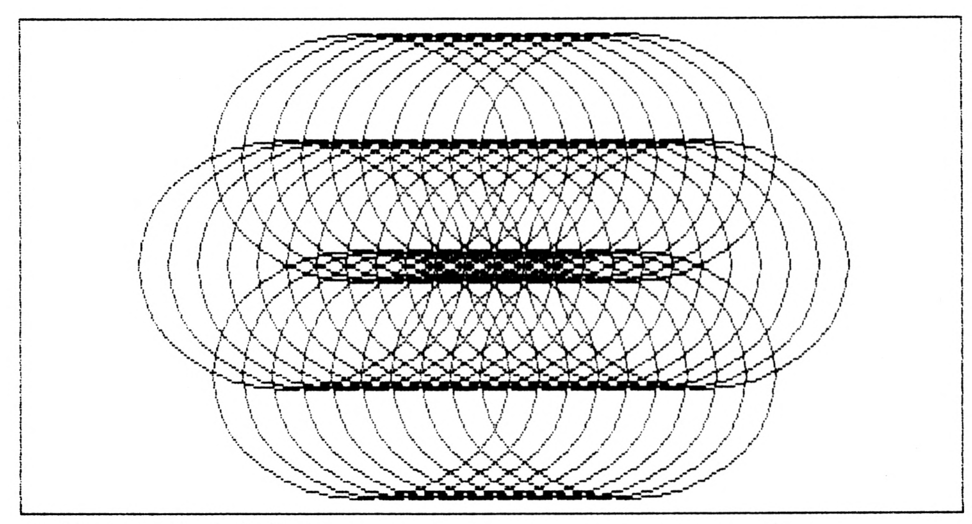
Grafik 21: Kreise mit veränderlichem Radius

```
300 REM--- Roehren-Grafik 2 ---
310 MODE 2:INK 0,0:INK 1,26
315 :
320 FOR t=4 TO 8.4 STEP 0.1
330 :   x0 = 340 +200*COS(t)
335 :   y0 = 340 -300*SIN(t/3)
340 :   h = t*t*t/4
360 :
370 REM--- MOVE x0-h,y0+h: DRAW x0+h,y0+h, 1
380 REM--- DRAW x0+h,y0-h, 1:DRAW x0-h,y0-h, 1
390 REM--- DRAW x0-h,y0+h, 1
395 :
400 :   ORIGIN x0, y0
405 :   DEG
410 :   FOR a=0 TO 360 STEP 10
415 :   x2 = h*COS(a): y2 = h*SIN(a)
420 :   IF a>0 THEN MOVE x1,y1:DRAW x2, y2, 1
430 :   x1 = x2: y1 = y2
440 :   NEXT a
450 :   RAD
460 :
470 NEXT t
480 :
490 a$=INKEY$: IF a$="" THEN 490
```

Beim nächsten Programm zum Zeichnen von Kreisen ist in Zeile 720 die Mittelpunktkoordinate y0, der Radius r und der Zentriwinkel w hinterlegt. Die x-Koordinate des Mittelpunktes läuft in der Schleife der Zeile 740 von 280 bis 360 mit der Schrittweite 20. Die Schleife in Zeile 770 veranlaßt das Zeichnen eines Kreises mit dem Mittelpunkt x0,y0, der Schrittweite 10°, dem Radius r und dem Farbstift 1.

Dieser Kreis (Schleife in Zeile 770) ist der Mittelpunkt von 6 weiteren Kreisen, deren Anzahl durch die Schleife in Zeile 820 festgelegt ist (s. auch w in Zeile 720). Die Zeilen 830,840 berechnen die x- und y-Koordinaten des Mittelpunktes (x,y) in Abhängigkeit von dem Schleifenzähler. Anschließend wird um den neu errechneten Mittelpunkt ein Kreis mit dem Radius r gezeichnet (Zeile 860–900).

```
700 REM--- Kreise zeichnen ---
710 MODE 2:BORDER 3:INK 0,0:INK 1,26
720 w = 60: y0 = 200: r = 100
725 :
730 DEG
735 :
740 FOR  x0 = 280 TO 360 STEP 20
```



Grafik 22: Geschachtelte Kreise

```
745 :
750 :
760 ORIGIN x0,y0
770 FOR a=0 TO 360 STEP 10
780 :   x2 = r*COS(a):y2 = r*SIN(a)
790 :   IF a>0 THEN
        MOVE x1, y1: DRAW x2, y2, 1
800 :   x1 = x2: y1 = y2
810 NEXT a
815 :
820 FOR n=1 TO 6
825 :
830 :   x = x0 + r*COS(n*w)
840 :   y = y0 + r*SIN(n*w)
845 :
850 :   ORIGIN x, y
860 :   FOR a=0 TO 360 STEP 10
870 :      x2 = r*COS(a):y2 = r*SIN(a)
880 :      IF a>0 THEN
           MOVE x1, y1:DRAW x2, y2, 1
890 :      x1 = x2: y1 = y2
900 :   NEXT a
905 :
910 NEXT n
```

```
920 :
925 :
940 NEXT x0
945 :
950 RAD
955 :
960 a$=INKEY$: IF a$="" THEN 960
```

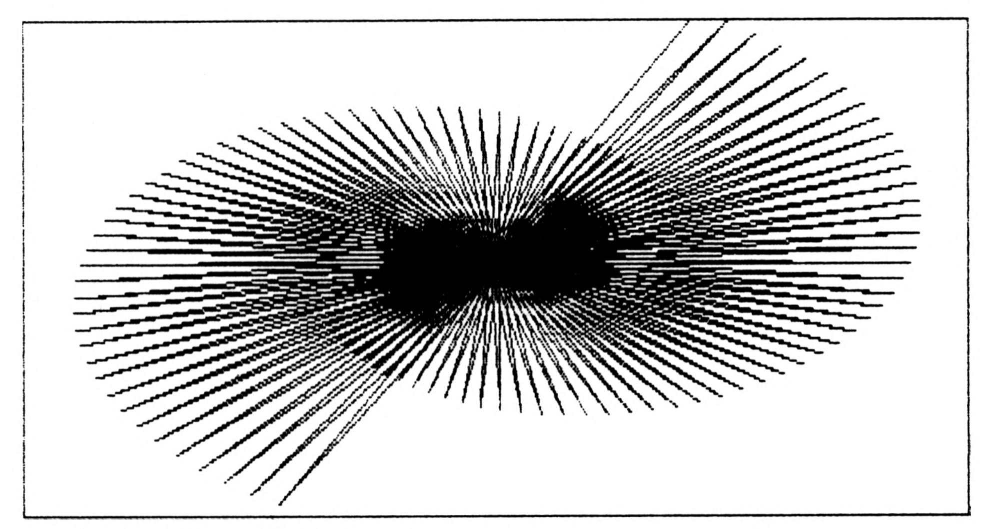
9.6 Turtle-Grafik

Schöne Muster und Grafiken lassen sich mit dem folgenden Programm generieren. Überraschend ist die Vielfalt von künstlerisch ansprechenden Bildern, die bei einer Parametervariation entstehen. Die Programmiersprache LOGO ermöglicht in einfachster Weise das Programmieren von solchen Turtle-Grafiken. Hierzu ist das Buch „Turtle Geometrie, the Computer as a Medium for Exploring Mathematics" von Harold Abelson und Andrea di Sesa (Cambridge Verlag, MIT – Press 1981) zu nennen.

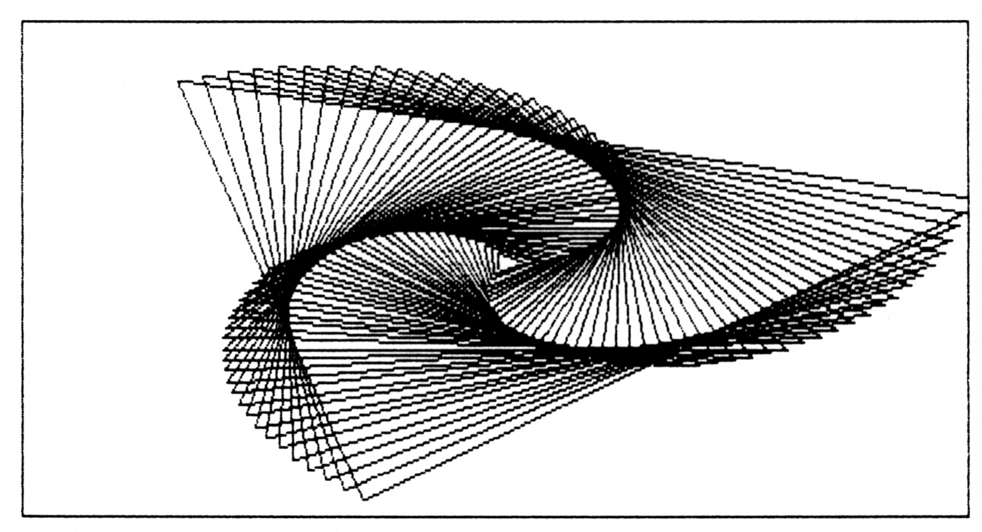
Um im folgenden Programm die Parameter sinnvoll variieren zu können, ist eine Erläuterung zur Turtle-Grafik erforderlich. Im Kapitel 9 „Parameterdarstellung der Ellipse" wurden die Gleichungen

```
x = x0 + a * COS(t)
y = y0 + b * SIN(t)
```

angegeben. Die Ellipsen-Halbmesser sind mit a,b und die Koordinaten des Ellipsen-Mittelpunktes mit x0,y0 bezeichnet. Die Verbindungsgerade vom Mittelpunkt (x0,y0) zum Ellipsenpunkt (x,y) nennen wir vereinfachend „Radius". Der „Radius" bildet mit der x-Achse



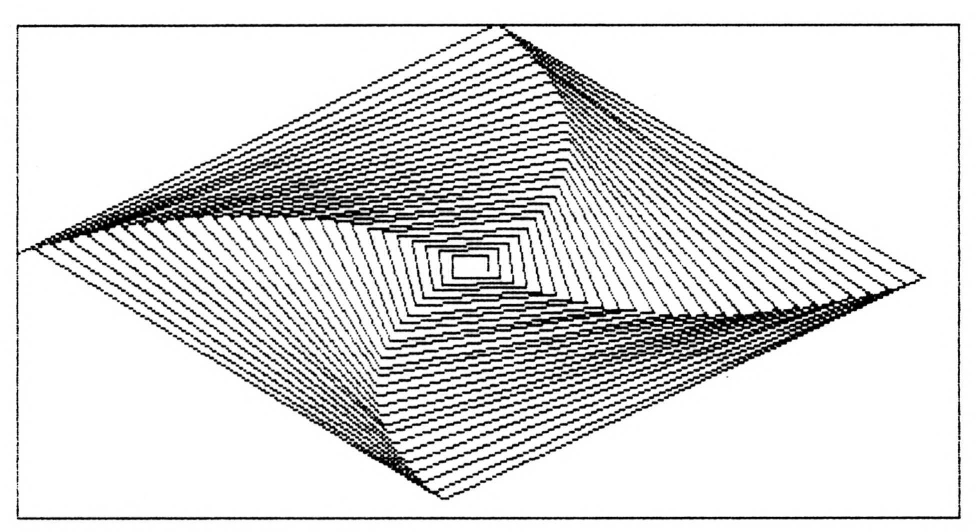
Grafik 23: Turtle-Grafik für td=182



Grafik 24: Turtle-Grafik für td=241

den Winkel t. Eine Umrechnung dieses Winkels in das Bogenmaß entfällt hier, da der Rechner durch die Zeile 660 auf Gradmaß eingestellt wird.

```
600 REM--- Turtle-Graphik ---
610 MODE 2:BORDER 9:INK 0,0:INK 1,18
620 a =20: b =a/1.6: ad =4: bd =ad/1.6
630 td = 182: t = 90
640 x0 = 320: y0 = 200
650 :
660 DEG
665 :
670 x = x0 + a * COS(t)
680 y = y0 + b * SIN(t)
690 :
700 WHILE x>0 AND x<639 AND y>0 AND y<399
710 :   x = x0 + a * COS(t)
720 :   y = y0 + b * SIN(t)
730 :   MOVE x0, y0: DRAW x, y, 1
740 :   x0 = x: y0 = y
750 :   t = t + td
760 :   a = a + ad
770 :   b = b + bd
780 WEND
790 :
800 RAD
```



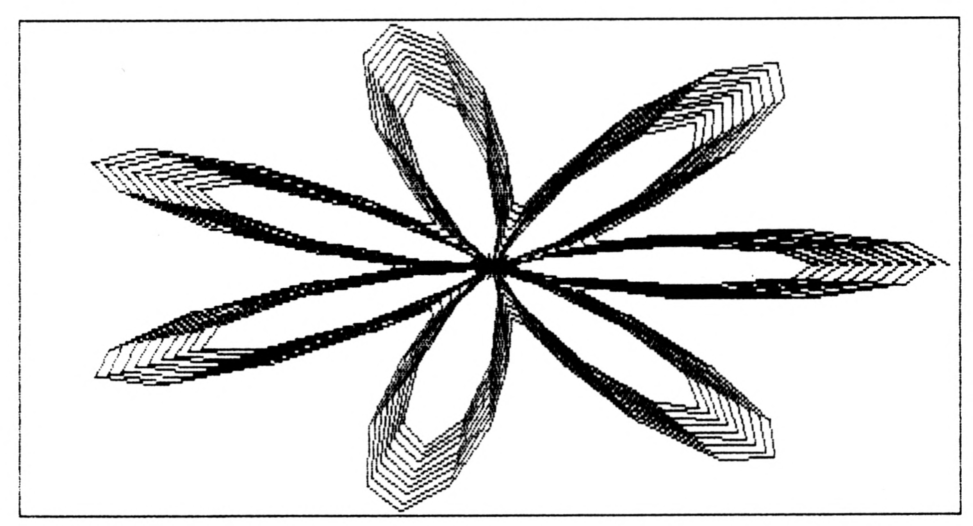
Grafik 25: Turtle-Grafik für td=90.4

```
805 :
810 a$=INKEY$: IF a$="" THEN 810
```

In der Zeile 740 wird der Ellipsenpunkt (x,y) als neuer Ellipsen-Mittelpunkt (x0,y0) verwendet. Der Winkel t zwischen dem „Radius" und der x-Achse wird in der Zeile 750 um den Winkel td erhöht. Der Parameter td beschreibt die Winkeländerung des „Radius". Die angegebenen Grafiken entstanden durch unterschiedliche Eingaben (Zeile 630) von td bei sonst ungeändertem Programm. In den Zeilen 760,770 werden die Halbachsen a,b um ad,bd vergrößert. Diese Vergrößerungen werden so lange ausgeführt, bis ein zu zeichnender „Radius"-Endpunkt (x,y) außerhalb des Bildschirmes liegt (Zeile 700).

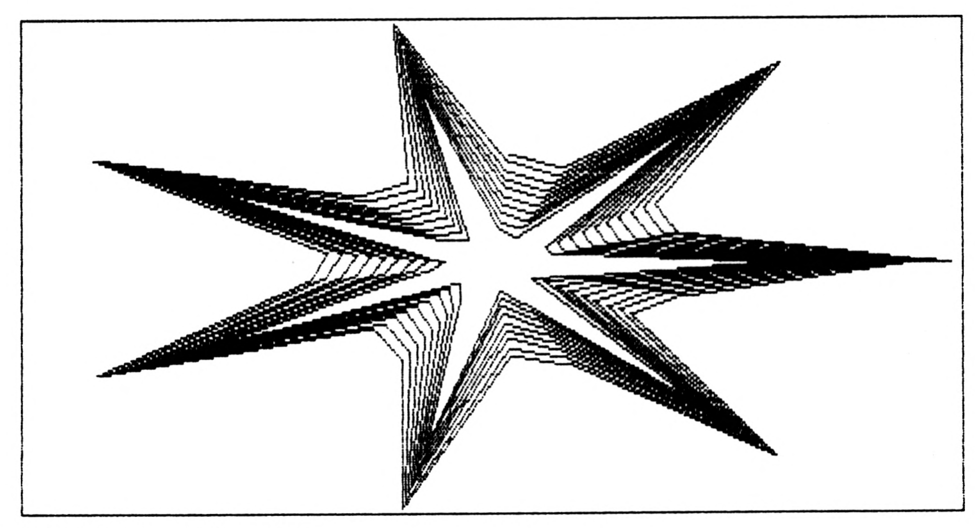
Das folgende Programm (Zeilen 800 ff.) ist ähnlich aufgebaut und ermöglicht z. B. das Zeichnen von Sternen. In den Zeilen 870,880 sind die Formeln der Parameterdarstellung einer Ellipse enthalten. Die Halbachsen a,b sind nicht konstant, sondern werden in der Zeile 860 verändert.

```
800 REM--- Stern-Grafiken ---
810 MODE 2:BORDER 9:INK 0,0:INK 1,18
820 td = 0.1: n = 7
830 x0 = 320: y0 = 200: b = 120
835 :
840 FOR a=100 TO x0-b STEP (x0-b)/20
850 :   FOR t=0 TO 6.4 STEP td
860 :   r = a + b*COS(n*t)
870 :   x = x0 + r*COS(t)
880 :   y = y0 + r*SIN(t)/1.5
890 :   IF x<0 OR x>639 OR y<0 OR y>399 THEN 950
```



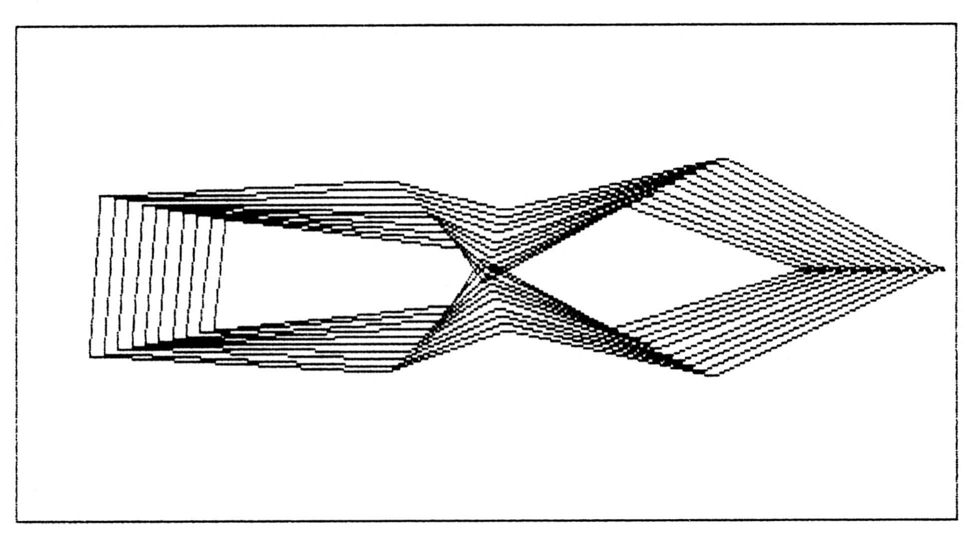
Grafik 26: „Blume" für td=0.1

```
900 :  IF t<>0 THEN MOVE x1, y1: DRAW x, y, 1
910 :  x1 = x: y1 = y
920 :  NEXT t
930 NEXT a
940 :
950 a$=INKEY$: IF a$="" THEN 950
```



Grafik 27: „Stern" für td=0.3

Die Schleifengrenzen in der Zeile 850 bewirken, daß (vereinfacht dargestellt) der „Radius" n Umläufe ausführt. Dabei erreichen die Halbachsen a,b (Zeile 860) n mal ihren Extremwert. Die Bilder 26,27,28 entstanden durch Variation von td (Zeile 820). Interessante Figuren ergeben sich z. B. auch für n=2,n=4,n=7.

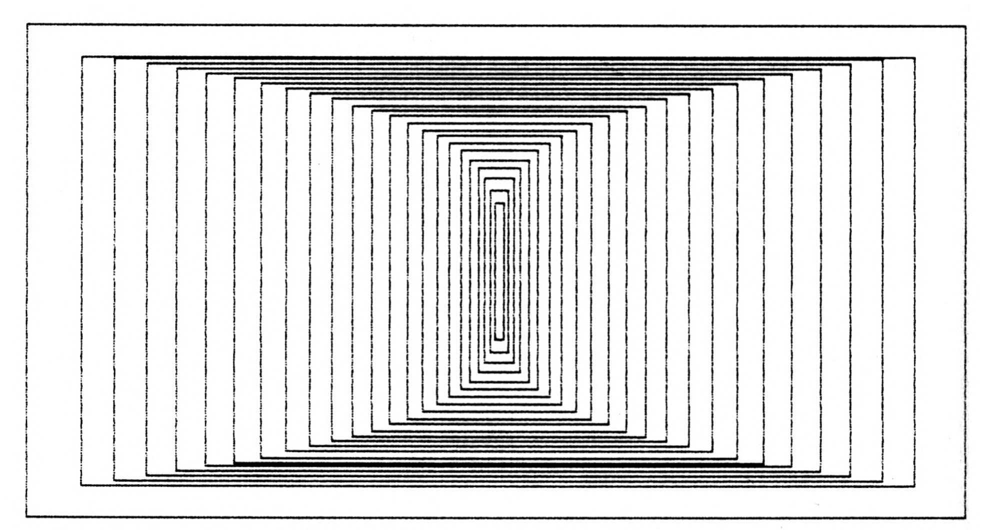


Grafik 28: „Pfeil" für td=0.7

9.7 Demo-Grafik mit Rechtecken

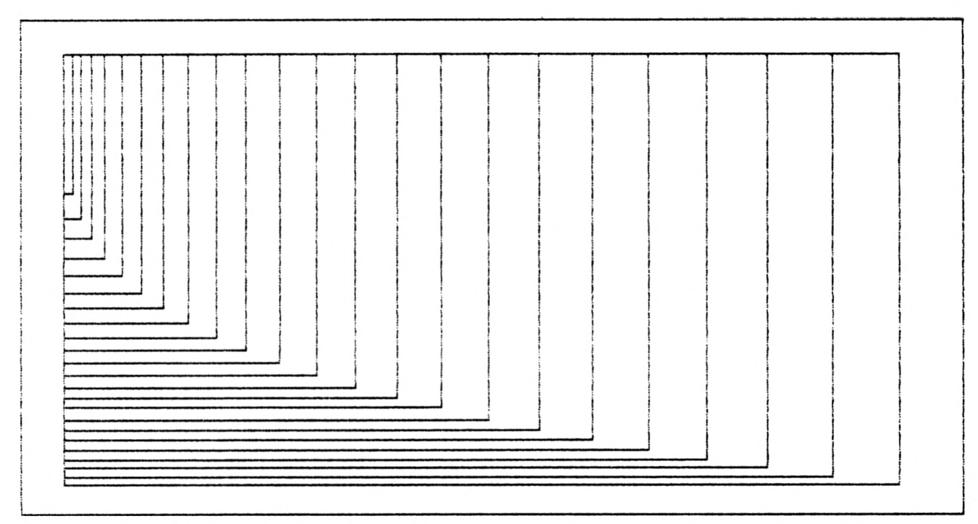
Durch das folgende Programm werden mittels einfacher Anweisungsfolgen geschachtelte Rechtecke auf den Bildschirm gezeichnet. Dabei wird zunächst der hochauflösende Bildschirm und die Rahmenfarbe violett (5) eingeschaltet (Zeile 510). In der gleichen Zeile wird die Hintergrundfarbe auf grün (9) und die Zeichenfarbe auf weiß (26) gesetzt. Die Zeile 520 bestimmt die Anzahl und den Abstand der in den Zeilen 550 bis 570 zu zeichnenden Rechtecke. Die x- und y-Ausdehnung der Rechtecke wird in der Zeile 540 festgelegt. Wird der Wert 35 des Ausdruckes y=35*sqr(t) erhöht, so erfolgt eine Ausdehnung in y-richtung.

```
500 REM--- Rechtecke ---
510 MODE 2:BORDER 5:INK 0,9:INK 1,26
515 :
520 FOR t=10 TO 100 STEP 4
530 :
```

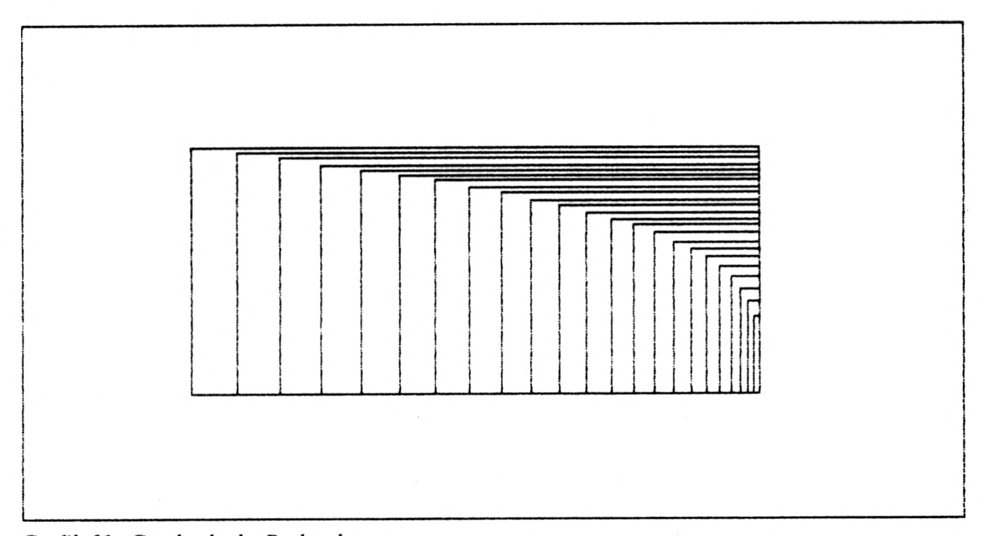


Grafik 29: Geschachtelte Rechtecke

```
540 :   y = 35 * SQR(t): x = t*t/17
550 :   MOVE 320-x/2, 200+y/2
555 :   DRAW 320+x/2, 200+y/2, 1
560 :   DRAW 320+x/2, 200-y/2, 1
565 :   DRAW 320-x/2, 200-y/2, 1
570 :   DRAW 320-x/2, 200+y/2, 1
580 :
590 :   REM--- y = 20 * SQR(t): x = t*t/25
600 :   REM--- MOVE 500-x, 100+y
605 :   REM--- DRAW 500  , 100+y, 1
610 :   REM--- DRAW 500  , 100  , 1
615 :   REM--- DRAW 500-x, 100  , 1
620 :   REM--- DRAW 500-x, 100+y, 1
630 :
640 :   REM*** y = 35 * SQR(t): x = t*t/17
650 :   REM*** MOVE 30+x, 370-y
655 :   REM*** DRAW 30  , 370-y, 1
660 :   REM*** DRAW 30  , 370  , 1
665 :   REM*** DRAW 30+x, 370  , 1
670 :   REM*** DRAW 30+x, 370-y, 1
680 :
690 NEXT t
700 a$=INKEY$ : IF a$="" THEN 700
```



Grafik 30: Geschachtelte Rechtecke



Grafik 31: Geschachtelte Rechtecke

Dies gilt in umgekehrter Form auch für den Wert 17 der x-Gleichung (Zeile 540). Vom Mittelpunkt (320,200) der Figur werden sich vergrößernde Rechtecke (s. Grafik 29) gezeichnet (Zeilen 550–570). Aktivieren wir die Zeilen 640 bis 670 und deaktivieren 540 bis 570, so werden zu den Koordinaten der linken oberen Ecke (30,370) die x-Ausdehnung addiert und die y-Ausdehnung subtrahiert. Es werden also von der linken oberen Ecke sich vergrößernde Rechtecke gezeichnet (Grafik 30). Aktivieren wir die Zeilen 590 bis 620, so wird der Festpunkt nach rechts unten (500,100) verschoben und die x- und y-Ausdehnung der Rechtecke verringert.

9.8 Eingeschriebene Vierecke

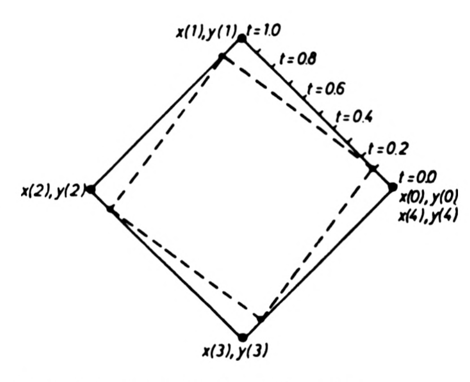


Abb. 13: Parameterdarstellung der Verbindungsgeraden von P1 *nach* P1

Zur Erklärung der folgenden Demo-Grafik betrachten wir das Viereck der Abb. 13. Die Koordinaten der Eckpunkte x(0),y(0), x(1),y(1), x(2),y(2), x(3),y(3) seien bekannt. Durch die Anweisungen MOVE x(i),y(i): DRAW x(i+1),y(i+1),1 wird eine Linie vom i-ten zum (i+1)-ten Eckpunkt gezeichnet (Zeile 670). Damit z. B. die Schleife

```
665 FOR i=0 TO 3
670 MOVE x(i),y(i):
    DRAW x(i+1),y(i+1),1
710 NEXT i
```

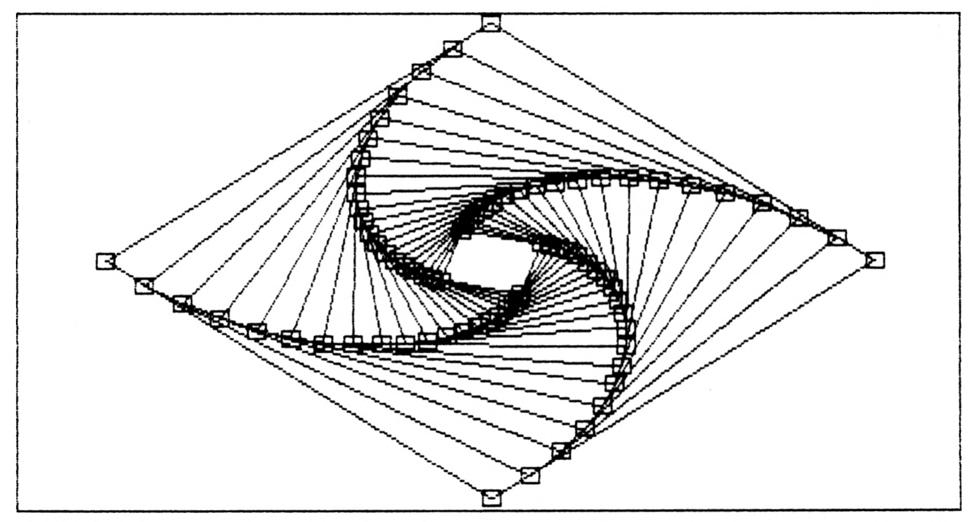
ein vollständiges Viereck zeichnet, ist vorher

```
x(4) = x(0), y(4) = y(0)
```

zu setzen (Zeile 715). Mit Hilfe der Parameterdarstellung der Geraden

```
x = x(i) + t * (x(i+1) - x(i))
y = y(i) + t * (y(i+1) - y(i))
```

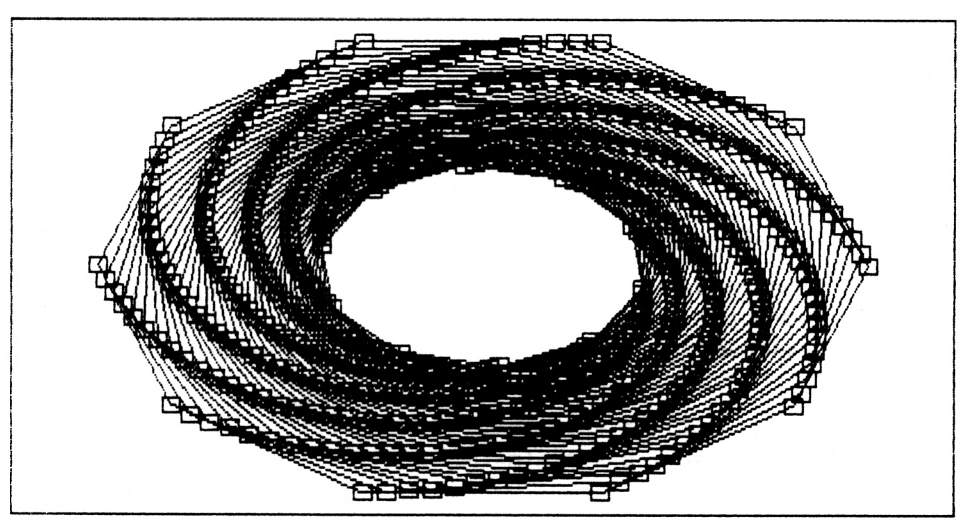
kann auf der Verbindungslinie zwischen dem i-ten und (i+1)-ten Eckpunkt ein Zwischenpunkt gewählt werden. Innere Teilungspunkte ergeben sich, wenn t zwischen 0 und 1 liegt (Zeile 655). In den Zeilen 700,705 werden die Koordinaten dieser inneren Punkte berech-



Grafik 32: Eingeschriebene Vierecke (n=4)

net und als neue Eckpunkte in die Variablen x(i),y(i) gespeichert. Beim Zeichnen ergibt sich eine Folge von eingeschriebenen Vierecken, deren Ecken auf den Seiten des vorher gezeichneten Viereckes liegen. Durch die Zeilen 675 bis 685 wird der jeweilige Endpunkt der gezeichneten Linie mit einem kleinen Rechteck markiert.

Für die erstmalige Besetzung der Eckpunkte wird die Parameterdarstellung der Ellipse (Zeile 635,640) benutzt. Die Werte von a,b (Zeile 610) bestimmen die Breite und Höhe der Grafik. Wird z. B. a kleiner gewählt, so wird die Figur in x-Richtung gestaucht. Der Mittelpunkt der Figur ist durch x0,y0 (Zeile 610) festgelegt. Der Wert n=4 (Zeile 615) legt fest, daß es sich um ein Viereck handelt. Die beiden folgenden Grafiken (32,33) ergeben sich für n=4 bzw. n=10.



Grafik 33: Eingeschriebene Zehnecke (n=10)

```
600 REM--- Eingeschriebene Vierecke ---
605 MODE 2:BORDER 2:INK 0,1:INK 1,24
610 x0 = 320: y0 = 200: a = 260: b = 190
615 n = 4: w0 = 2*PI/n
620 :
625 FOR i=0 TO n
630 :   t = i * w0
635 :   x(i) = x0 + a*COS(t)
640 :   y(i) = y0 + b*SIN(t)
645 NEXT i
650 :
655 t = 0.1
660 FOR j=1 TO 5*n
```

```
665 : FOR i=0 TO n-1
670 :    MOVE x(i), y(i): DRAW x(i+1), y(i+1), 1
675 :    MOVE x(i+1)-6, y(i+1)+6
677 :    DRAW x(i+1)+6, y(i+1)+6, 1
680 :    DRAW x(i+1)+6, y(i+1)-6, 1
682 :    DRAW x(i+1)-6, y(i+1)-6, 1
685 :    DRAW x(i+1)-6, y(i+1)+6, 1
700 :    x(i) = x(i) + t * ( x(i+1) - x(i) )
705 :    y(i) = y(i) + t * ( y(i+1) - y(i) )
710 : NEXT i
715 : x(n) = x(0): y(n) = y(0)
720 NEXT j
722 :
725 a$=INKEY$: IF a$="" THEN 725
```

9.9 Lissajous-Figuren

LISSAJOUS-Figuren sind ebene Kurven, die durch die Überlagerung zweier in unterschiedlicher Richtung erfolgender Schwingungen entstehen. Wir bezeichnen die Frequenzen der Schwingungen in x- bzw. in y-Richtung mit n bzw. m und beschreiben diese Schwingungen durch

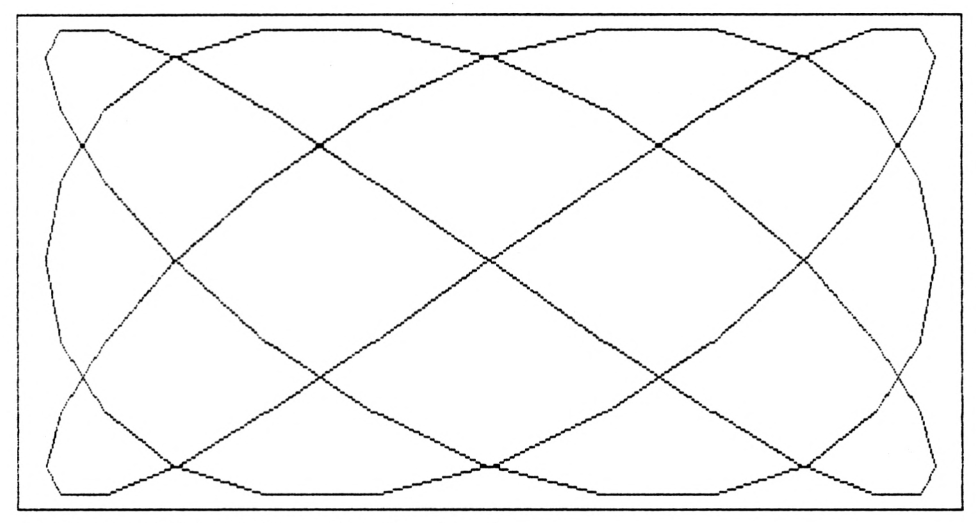
```
x = x0 + a * cos(n*t)
y = y0 + b * sin(m*t).
```

Hierbei sind a,b die x,y-Amplituden der Schwingungen. Der Parameter t beschreibt z. B. die Zeit.

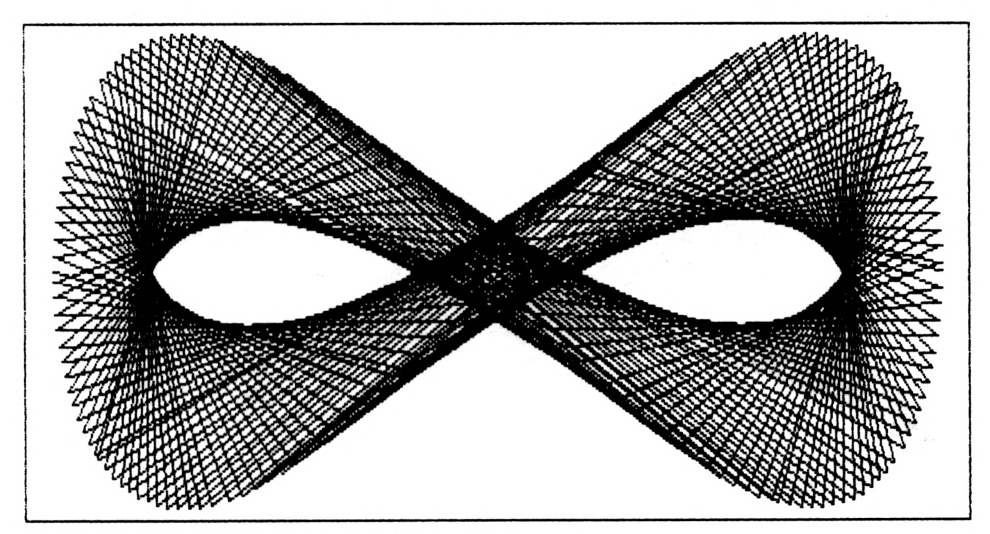
In der Zeile 835 des folgenden Programmes zum Zeichnen von LISSAJOUS-Figuren wird lediglich eine Umrechnung des Winkels w ins Bogenmaß durchgeführt. Schreitet die Zeit 1 fort, so bilden bei ganzzahligen Werten von n,m diese Kurven geschlossene Linien. Als Beispiel für eine LISSAJOUS-Figur ist für n=3,m=4 und der Schrittweite d=5 (Grafik 34) eine solche geschlossene Linie dargestellt.

```
800 REM --- Lissajous-Figuren ---
810 x0 = 320: y0 = 200: a = 300: b = 190
815 d = 5: n = 3: m = 4: t0 = PI/180
820 MODE 2:BORDER 2:INK 0,3:INK 1,26
825 :
830 FOR w=0 TO 180*d STEP d
835 :   t = w * t0
840 :   x = x0 + a*COS(n*t)
845 :   y = y0 + b*SIN(m*t)
```

```
860 :   IF t<>0 THEN MOVE x1, y1: DRAW x, y, 1
870 :   x1 = x: y1 = y
880 NEXT w
885 :
890 a$=INKEY$: IF a$="" THEN 890
```

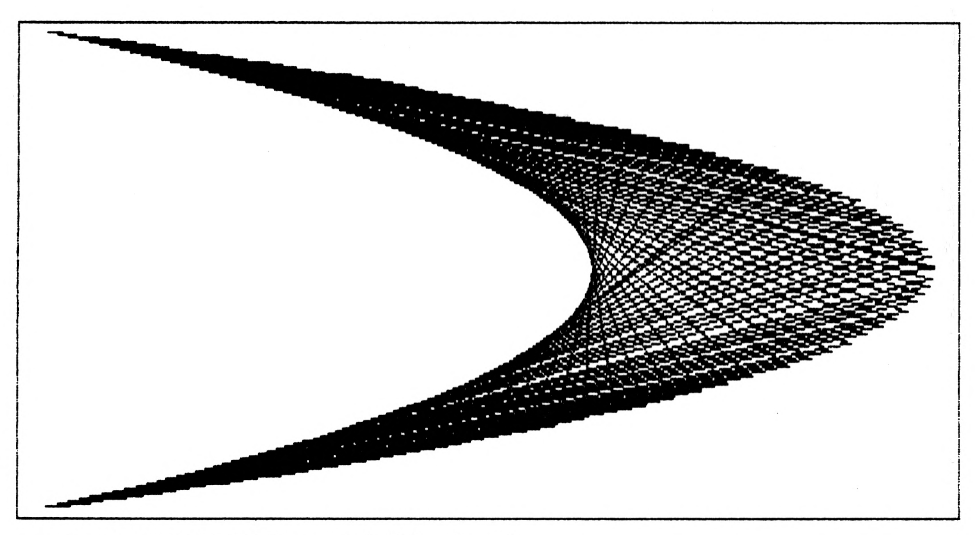


Grafik 34: LISSAJOUS-Figur mit d=5,n=3,m=4

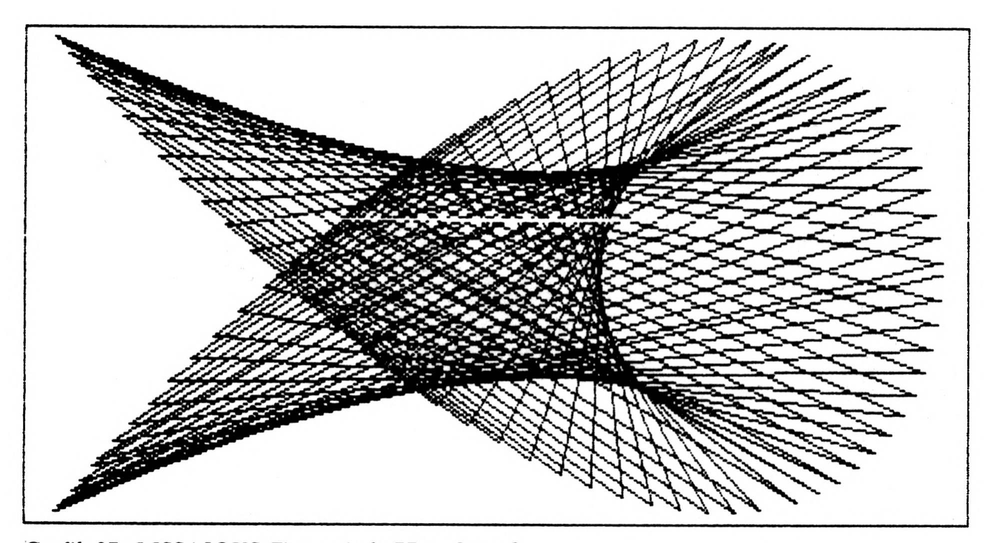


Grafik 35: LISSAJOUS-Figur mit d=77,n=1,m=2

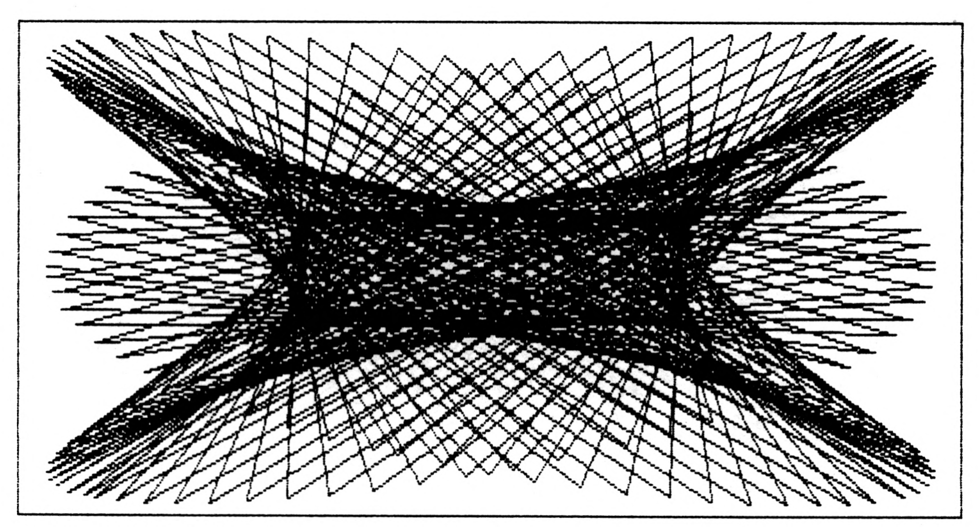
Wird die Schrittweite d verändert, so werden weiter auseinanderliegende Kurvenpunkte durch Geraden verbunden. Durch Variation von n,m ergeben sich dann die vielfältigsten Grafiken (s. Grafiken 35 bis 42).



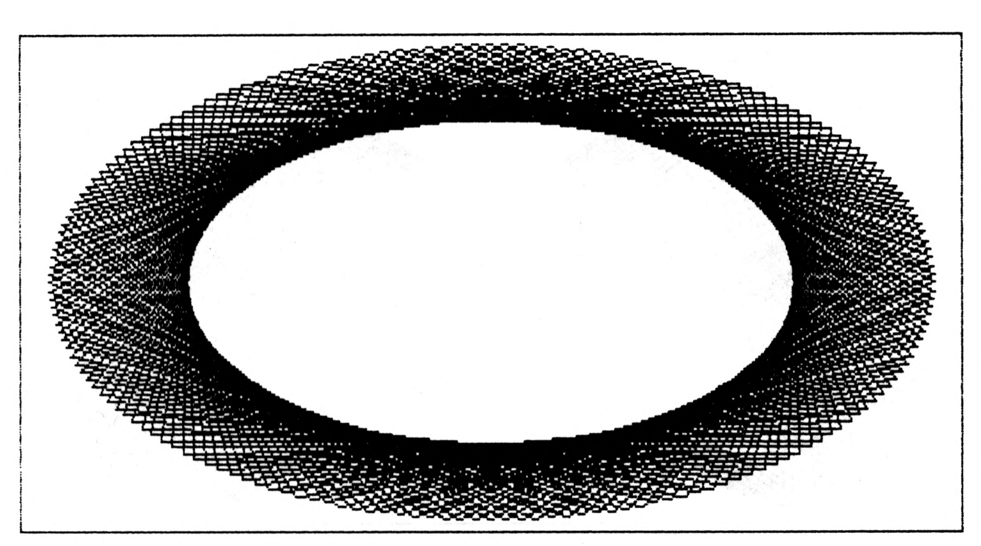
Grafik 36: LISSAJOUS-Figur mit d=77,n=2,m=1



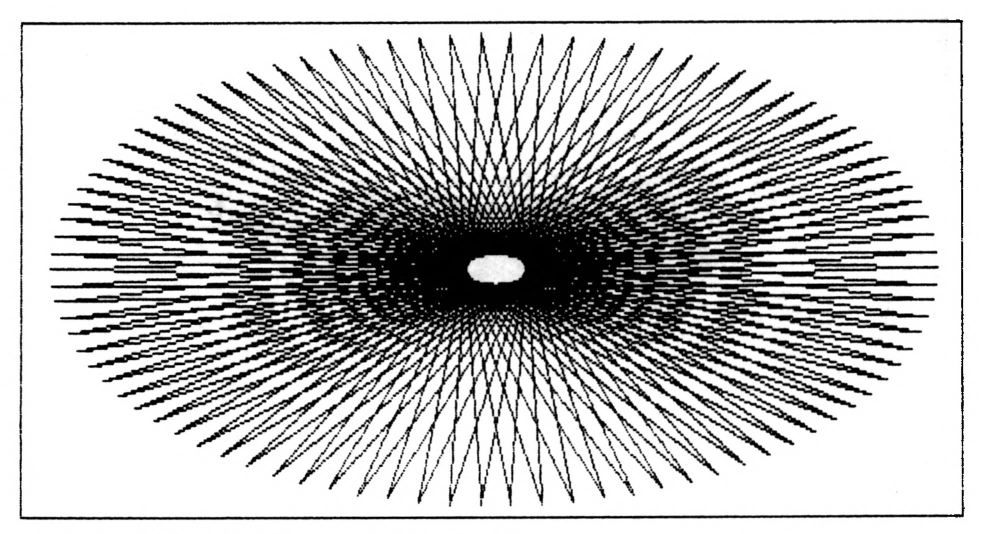
Grafik 37: LISSAJOUS-Figur mit d=77,n=2,m=3



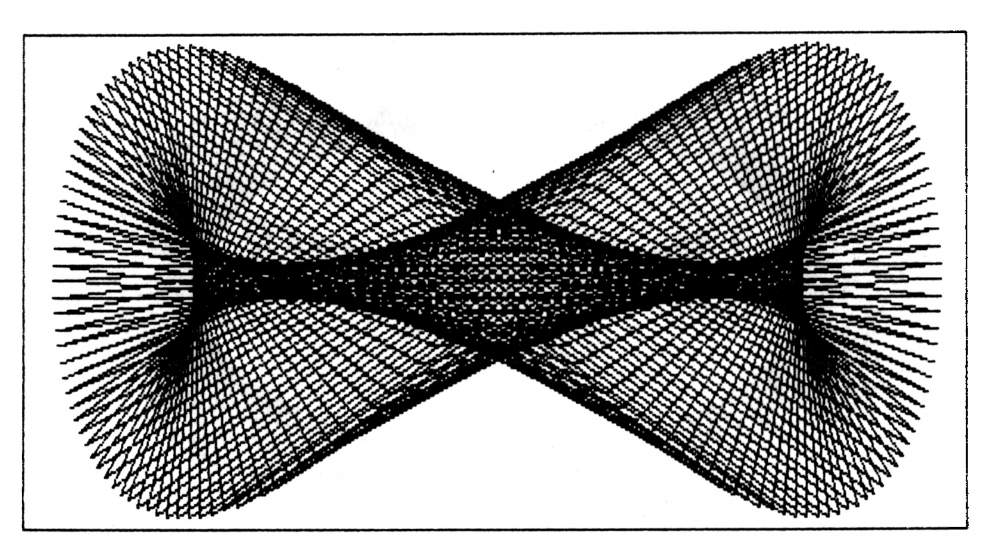
Grafik 38: LISSAJOUS-Figur mit d=77,n=3,m=2



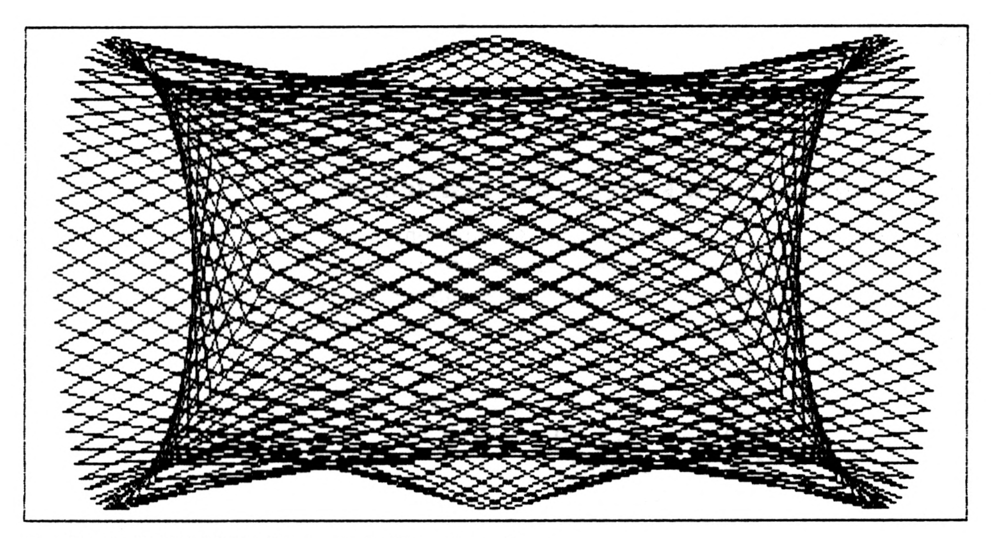
Grafik 39: LISSAJOUS-Figur mit d=94,n=1,m=1



Grafik 40: LISSAJOUS-Figur mit d=94,n=2,m=2



Grafik 41: LISSAJOUS-Figur mit d=94,n=1,m=2



Grafik 42: LISSAJOUS-Figur mit d=94,n=1,m=3

10. Zeichnen von Funktionen

10.1 Funktionen in kartesischer Darstellung

Funktionen stellen eine mathematische Beschreibung von gegenseitigen Abhängigkeiten dar. In allen wissenschaftlichen Disziplinen werden Funktionen zur Beschreibung dieser Abhängigkeiten genutzt. Wir gehen davon aus, daß eine Funktion y=y(x) als Formel vorgegeben ist, wie z. B. y=sin(x). Für eine solche Funktion kann man eine Wertetabelle aufstellen. Durch das Programm

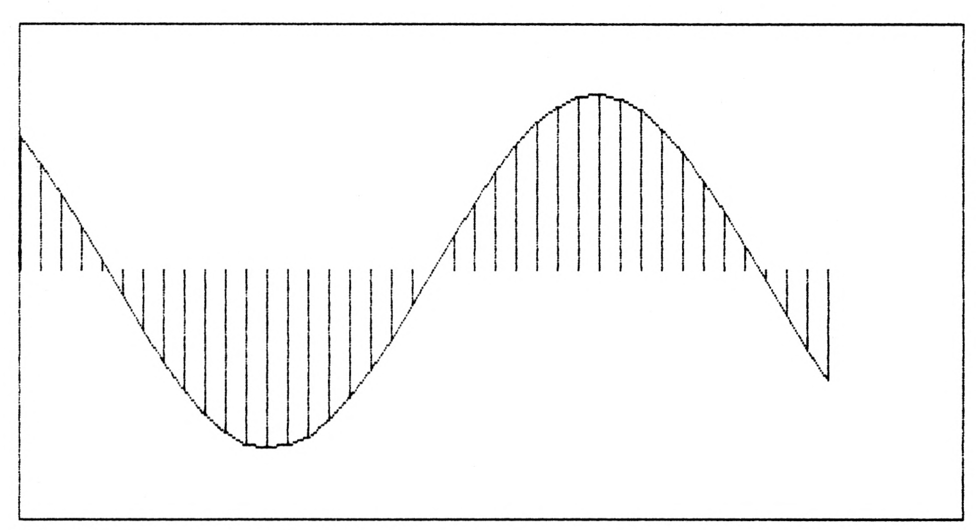
```
120 x8 =-4: x9 = 4.0: dx = 0.2
150 FOR x=x8 TO x9 STEP dx
160 y = SIN(x)
240 PRINT x, y
260 NEXT x
```

wird zu dem jeweiligen Wert x der Funktionswert y (Zeile 160) berechnet und ausgegeben (Zeile 240). In der Zeile 120 sind die x-Grenzen x8,x9 und die Schrittweite dx hinterlegt. Im Gegensatz zur Wertetabelle ermöglicht die grafische Darstellung einer Funktion einen umfassenden Überblick. Daher werden grafische Darstellungen funktionaler Zusammenhänge vielfach bevorzugt angewendet. Zum Auftragen der Wertepaare x,y auf dem Bildschirm ist die Umrechnung von x,y in die physikalischen Bildschirmkoordinaten (0<=i%<=639,0<=j%<=399) erforderlich.

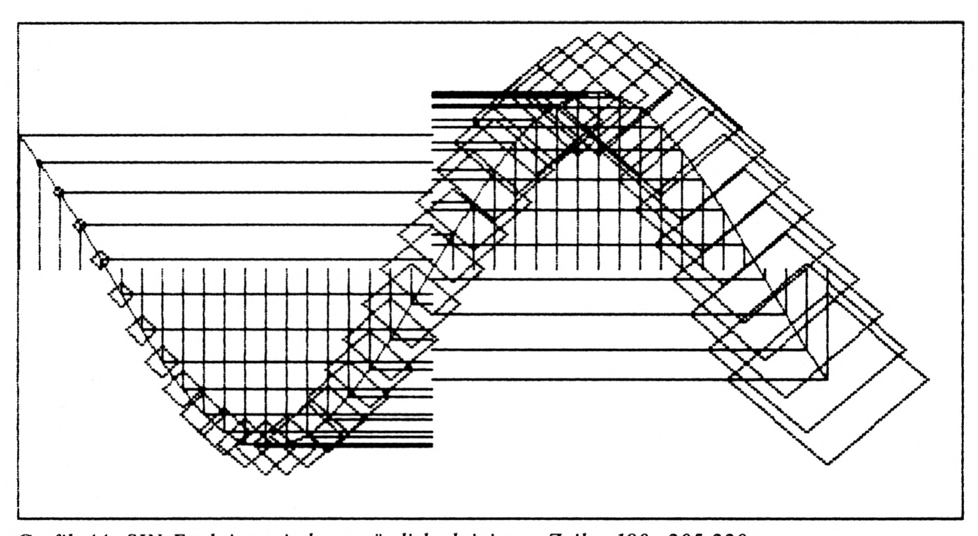
In der Zeile 140 des folgenden Programmes wird der Umrechnungsfaktor mx mit Hilfe des kleinsten (x8) und größten (x9) x-Wertes und my (Zeile 145) mit Hilfe des kleinsten (y8) und größten (y9) y-Wertes ermittelt. Da in der Zeile 130 i9=560.5 und nicht i9=639.5 verwendet wird, wird mx (Zeile 140) zu klein. Die Kurve (siehe Grafik 43) erreicht nicht den rechten Bildrand. In den Zeilen 170,175 wird der zu x,y gehörende Pixel (Bildpunkt) i%,j% berechnet. In der Zeile 240 wird eine Linie von i1,j1 nach i%,j% gezeichnet. Der Wert dx wurde mit 0.2 hinreichend klein gewählt.

Deshalb vermitteln die aufeinanderfolgenden Sehnenzüge (Zeile 240) ein „gerundetes" Bild des Kurvenverlaufes für x=sin(x). Zusätzlich können wir jeweils an der Stelle i% eine senkrechte Linie vom Kurvenpunkt (i%,j%) zur Nullinie (y=0) zeichnen lassen (s. j0,Zeile 145,230). Entsprechendes gilt für x0=0 (s.i0,Zeile 140,220). Wir erhalten Grafik 43. Aktivieren wir die Zeilen 190 bis 205, indem wir „REM" löschen, so werden um den jeweiligen Mittelpunkt (i%,j%) Vierecke gezeichnet. Die Größe dieser Vierecke nimmt von links nach rechts zu (Grafik 44). Durch die aktivierte Zeile 210 werden alle Kurvenpunkte (i%,j%) mit dem Punkt P(x=x8,y=y0) durch Linien verbunden. In ähnlicher Weise sind natürlich weitere Programmänderungen möglich.

Sollen andere Funktionen grafisch dargestellt werden, so ist diese Funktion in der Zeile 160 einzugeben.



Grafik 43: Funktion y(x)=sin(x)



Grafik 44: SIN-Funktion mit den zusätzlich aktivierten Zeilen 190–205,220

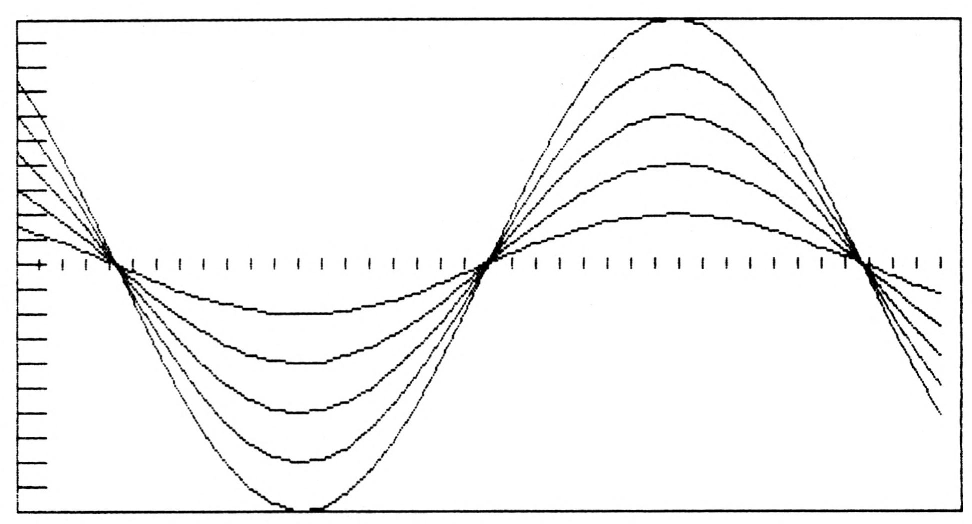
Damit die Kurve zwischen dem oberen und unteren Bildrand liegt, muß der größte (y9) und der kleinste (y8) Funktionswert y in der Zeile 125 hinterlegt werden. Ebenso ist in der Zeile 120 der gewünschte Zeichenbereich x8,x9 und die Schrittweite dx einzugeben.

```
100 REM---   sin()-Funktion zeichnen ---
110 MODE 2:BORDER 2:INK 0,1:INK 1,26
120 x8 =-4:   x9 = 4: dx = 0.2
```

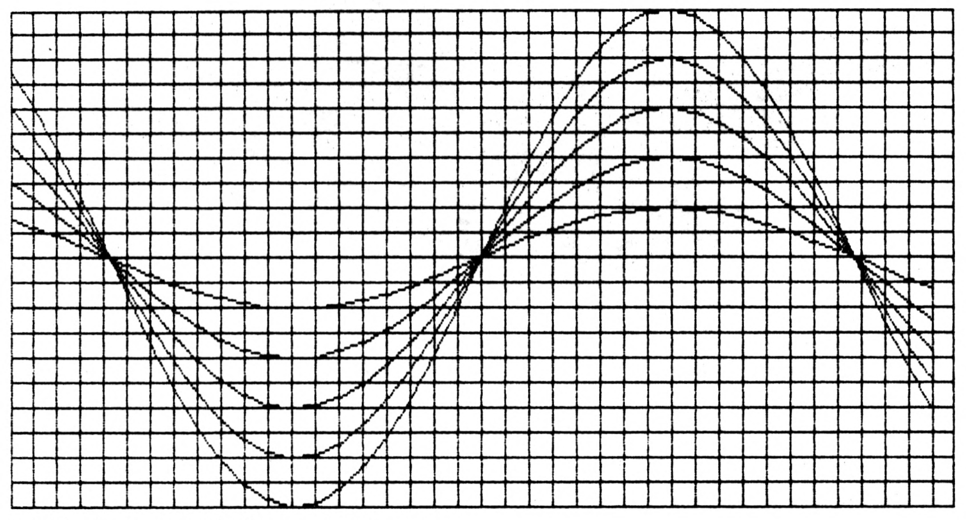
```
125 y8 =-1.4: y9 = 1.4: x0 = 0: y0 = 0
127 :
130 i8 = 0.5: i9 = 560.5
135 j8 = 0.5: j9 = 399.5
137 :
140 mx = (i9-i8)/(x9-x8): i0 = i8 + mx*(x0-x8)
145 my = (j9-j8)/(y9-y8): j0 = j8 + my*(y0-y8)
147 :
150 FOR x=x8 TO x9 STEP dx
160 :   y = SIN(x)
170 :   i% = i8 + mx * (x-x8)
175 :   j% = j8 + my * (y-y8)
177 :
190 REM+++ MOVE i%-i%/8,j%: DRAW i%,j%+i%/8, 1
200 REM+++ DRAW i%+i%/8,j%, 1: DRAW i%,j%-i%/8, 1
205 REM+++ DRAW i%-i%/8,j%, 1
210 REM*** MOVE i%,j%: DRAW i8,j0, 1
220 REM--- MOVE i%,j%: DRAW i0,j%, 1
225 :
230 :   MOVE i%,j%: DRAW i%,j0, 1
240 :   IF x<>x8 THEN MOVE il,jl: DRAW i%,j%, 1
250 :   il = i% : jl = j%
260 NEXT x
270 :
280 a$=INKEY$: IF a$="" THEN 280
```

10.2 Zeichnen von Parameterfunktionen

Als Ergänzung zum vorhergehenden Programm wollen wir nun parameterabhängige Funktionen x=y(x,a) grafisch darstellen. Für verschiedene Amplitudenwerte a soll z. B. die SINUS-Funktion y=a∗sin(x) gezeichnet werden. Solchen Funktionen entsprechen Schwingungen, deren maximale Auslenkungen durch a gegeben sind. Die Änderung des Parameters a geschieht durch die Schleife der Zeile 355. Hierbei läuft x jeweils von x8 bis x9 mit einer Schrittweite dx (Zeile 360). Für jeden a-Wert wird eine SINUS-Linie gezeichnet. Der mehrfache Durchlauf der a-Schleife liefert Kurvenscharen (s. Grafik 45). Damit alle Kurventeile auf dem Bildschirm gezeichnet werden, ist der kleinste (y8) und größte (y9) Ordinatenwert y (s. Zeile 325) ausreichend zu setzen. Damit wir aus der grafischen Darstellung der Funktion auch Zwischenwerte ablesen können, werden durch die beiden Schleifen 510 und 550 Achsenmarkierungen gezeichnet. Die Schrittweite in x-Richtung ist durch dx gegeben, d. h. die x-Achsenmarkierungen laufen von x8 bis x9 mit der Schrittweite dx. Die y-Achsenmarkierungen unterteilen den Bereich von y8 bis y9 in 20 Teile (Zeile 550).



Grafik 45: Parameterfunktion mit a = 2,4,6,8,10



Grafik 46: Parameterfunktionen im Gitternetz

Soll anstelle der kurzen Achsenstrichelchen (s. Grafik 45) ein durchlaufendes Gitternetz gezeichnet werden, so sind die Zeilen 520 bzw. 560 durch

```
520 MOVE i,j8: DRAW i,j9, 1
```

bzw.

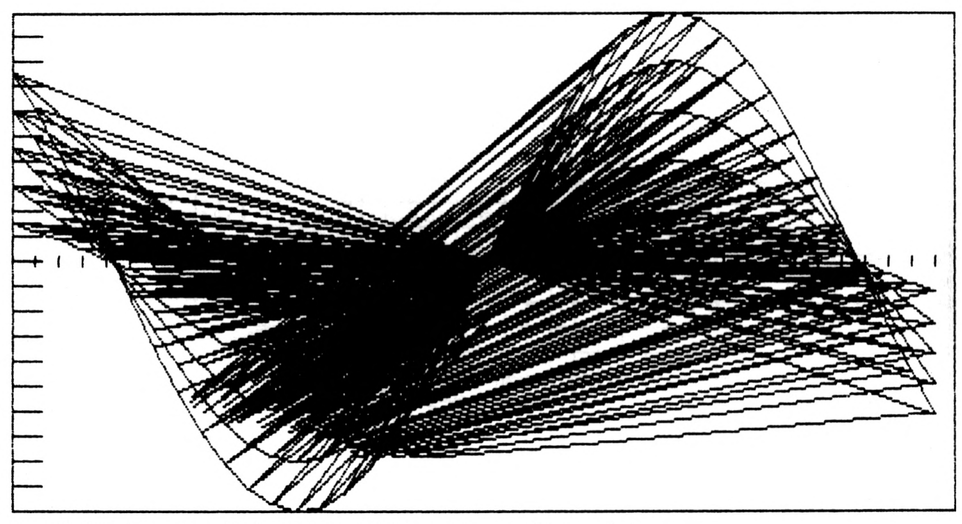
```
560 MOVE i8,j: DRAW i9,j, 1
```

zu ersetzen (Grafik 46).

Natürlich können diese Kurvenscharen für a=2,4,6,8,10 leicht durch zusätzliche Linien optisch modifiziert werden. Die aktivierte Zeile 400 (s. REM-Zeilen am Programmende) zeichnet jeweils eine zusätzliche Linie von jedem Kurvenpunkt (i%,j%) zum Punkt (x=0,y=0).

Mit den aktivierten Zeilen 400,410 ergibt sich die Grafik 47. Soll eine andere Parameterfunktion y=y(x,a) mit den Grenzen x8,x9 und der Schrittweite dx gezeichnet werden, so ist die Zeile 370 entsprechend zu ersetzen. Der zu variierende Parameter ist hierbei mit a zu bezeichnen. Beispiele für solche Funktionen sind:

```
370 y = EXP(-x/a) * COS(x)
370 y = a - x*x / a
370 y = 2 * x * a * (RND(1)-0.5)
```



Grafik 47: Parameterfunktion bei aktivierten Zeilen 400 und 410

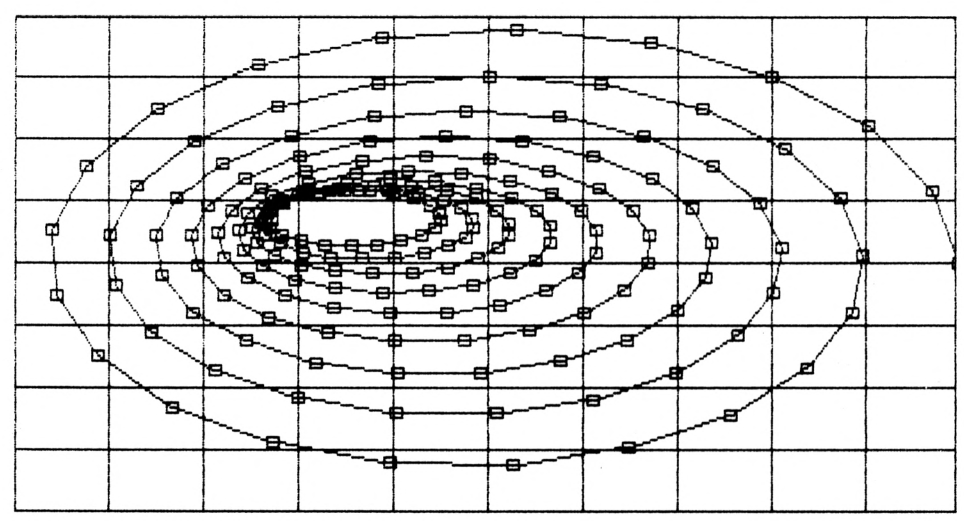
```
300 REM--- Parameter-Funktion zeichnen ---
310 MODE 2:BORDER 2:INK 0,1:INK 1,26
320 x8 = -4: x9 =  4: dx = 0.2
325 y8 =-10: y9 = 10
330 i8 = 0.5: i9 = 639.5
335 j8 = 0.5: j9 = 399.5
340 mx = (i9-i8)/(x9-x8)
345 my = (j9-j8)/(y9-y8)
350 :
355 FOR a=2 TO 10 STEP 2
360 :  FOR x=x8 TO x9 STEP dx
```

```
365 :
370 :     y = a * SIN(x)
375 :
380 :     i% = i8 + mx * (x-x8)
390 :     j% = j8 + my * (y-y8)
420 :     IF x<>x8 THEN MOVE il,jl: DRAW i%,j%, 1
430 :     il = i%: jl = j%
440 :   NEXT x
450 NEXT a
460 :
470 MOVE i8,j8: DRAW i9-i8,j8, 1
480 DRAW i9-i8,j9-j8, 1:DRAW i8,j9-j8, 1
490 DRAW i8,j8, 1
500 :
510 FOR i=i8 TO i9 STEP mx*dx
520 MOVE i,j8-my*y8+4: DRAW i,j8-my*y8-4, 1
530 NEXT i
540 :
550 FOR j=j8 TO j9 STEP (j9-j8)/20
560 MOVE i8,j: DRAW i8+20,j, 1
570 NEXT j
580 :
590 a$=INKEY$: IF a$="" THEN 590
600 END
610 :
620 REM** falls gewuenscht Programm-Zeilen --
630 REM** 400,410 wie folgt ergaenzen:     --
635 REM-- 400 MOVE i%,j%:
              DRAW i8-mx*x8,j8-my*y8, 1
640 REM-- 410 MOVE i%,j%:
              DRAW i8-mx*x8-j%/2,j8-my*y8-i%/4, 1
```

10.3 Zeichnen von Ortskurven

In der Elektrotechnik werden vielfach Zeiger in der komplexen Ebene zur Darstellung von sich zeitlich ändernden Strömen und Spannungen verwendet. Werden diese komplexen Funktionen in einen REAL-Teil x(t) und einen IMAGINÄR-Teil y(t) aufgespalten, so können die zugehörigen Ortskurven als Parameterdarstellung x(t),y(t) gezeichnet werden. Der Parameter t läuft von t8 bis t9 mit der Schrittweite dt (Zeile 610). Die min- bzw. max-x,y-Werte werden in den Zeilen 620,625 hinterlegt. Die Werte dx,dy (Zeilen 620,625) stellen

die Schrittweiten für das Gitternetz (Zeilen 755–785) dar. Die Parameterdarstellung x(t),y(t) der Ortskurve wird in den Zeilen 670,675 eingegeben. Durch die Zeilen 685,690 werden die x,y-Werte in die Bildschirmkoordinaten i%,j% umgerechnet. Zwei aufeinanderfolgende Punkte werden durch eine Linie (Zeile 700) verbunden. Jeder Punkt entspricht einem bestimmten t-Wert und wird durch ein kleines Rechteck (Zeilen 705–715) markiert.



Grafik 48: Ortskurve x(t),y(t)

```
600 REM--- Zeichnen von Ortskurven ---
605 MODE 2:BORDER 1:INK 0,2:INK 1,26
610 t8 = 0: t9 = 60: dt = 0.3
615 :
620 x8 = -1: x9 =  1: dx = 0.2
625 y8 = -1: y9 =  1: dy = 0.25
630 :
635 i8 = 0.5: i9 = 639.5
640 j8 = 0.5: j9 = 399.5
645 mx = (i9-i8)/(x9-x8)
650 my = (j9-j8)/(y9-y8)
655 :
660 FOR t=t8 TO t9 STEP dt
665 :
670 :   x = EXP(-0.03*t)*COS(t)  - t/200
675 :   y = EXP(-0.04*t)*SIN(t)  + t/300
680 :
```

```
685 :   i% = i8 + mx * (x-x8)
690 :   j% = j8 + my * (y-y8)
695 :
700 :   IF t<>t8 THEN MOVE i1,j1: DRAW i%,j%, 1
705 :   MOVE i%-4,j%+4: DRAW i%+4,j%+4, 1
710 :   DRAW i%+4,j%-4, 1: DRAW i%-4,j%-4, 1
715 :   DRAW i%-4,j%+4, 1
720 :   i1 = i%: j1 = j%
725 NEXT t
730 :
735 MOVE i8,j8: DRAW i9-i8,j8, 1
740 DRAW i9-i8,j9-j8, 1: DRAW i8,j9-j8, 1
745 DRAW i8, j8, 1
750 :
755 FOR i=i8 TO i9 STEP mx*dx
760 : MOVE i, j8: DRAW i, j9, 1
765 NEXT i
770 :
775 FOR j=j8 TO j9 STEP my*dy
780 : MOVE i8, j: DRAW i9, j, 1
785 NEXT j
790 :
800 a$=INKEY$: IF a$="" THEN 800
```

11. Präsentationsgrafiken

11.1 Balkendiagramme

Balkendiagramme und Blockgrafiken werden bei vielen Anwendungen benutzt, z. B.
- zur Darstellung der Wählerstimmen in bezug auf die Parteien
- zur grafischen Darstellung des Notenspiegels von Klassenarbeiten
- zur Darstellung des Umsatzes über der Abrechnungsperiode.

Wir wollen ein universelles Programm entwickeln, das mit dem BASIC-Standard-Befehlssatz auskommt und keine zusätzlichen Grafik-Befehle benötigt.

Im Beispiel wollen wir den monatlichen Umsatz (in Mill. DM) als Block auftragen. Der Wert des Umsatzes soll in die schwarz ausgefüllten Blöcke geschrieben werden (s. Abb. 14). Die Monatsnamen t$ und die dazugehörigen Umsätze x$ werden in DATA-Zeilen (Zeilen 120–170) als Zeichenketten abgelegt. Die Überschrift für das Balkendiagramm ist vor dem ersten Monatsnamen in der Zeile 110 enthalten. Das Datenende markieren die Zeichen ee (Zeile 180). Vor dem Datenende ee können beliebig viele Datenpaare vorhanden sein. Das Balkendiagramm wird so dargestellt, daß die Länge eines Balkens dem Umsatz entspricht.

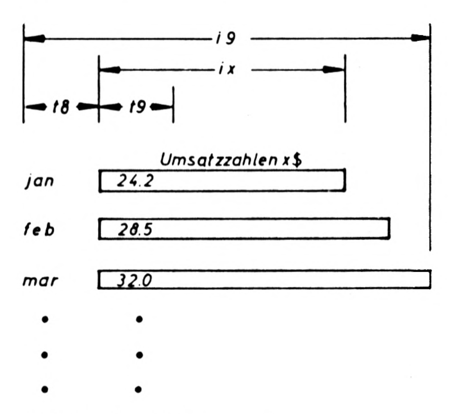
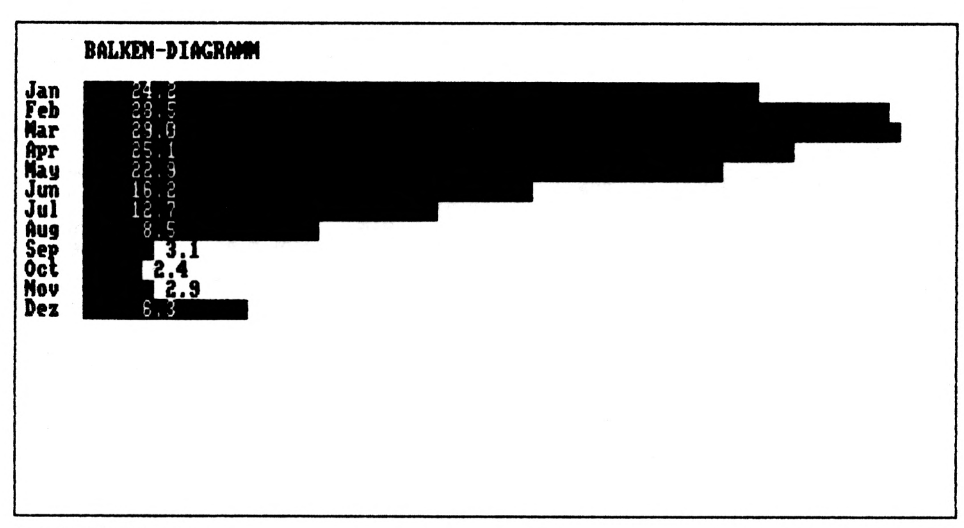


Abb. 14: Bezeichnungen für das Balkendiagramm

In dem Programmstück von 215 bis 280 werden die Daten nach dem kleinsten (x8) und größten Umsatz (x9) durchsucht und die größte Zeichenzahl t8 aller t$ (bzw. t9 aller x$) ermittelt. In der Zeile 300 wird t8 aufgrund der beiden Blanks (Leerzeichen) hinter dem Monatsnamen vergrößert und ebenso t9 wegen der Blanks vor den Umsatzzahlen. In der Variablen n (Zeile 230) wird die Anzahl der gelesenen Datenpaare hochgezählt, bis das Datenende ee erreicht ist. Der Maßstabsfaktor mx wird in der Zeile 310 ermittelt, indem die zur Verfügung stehende Länge (i9–t9) durch die Umsatzspannweite (x9–x8) dividiert wird. In der Zeile 330 wird die Zeichenkette b$ mit ausreichend vielen Blanks gefüllt. Das erste Zeichen (CHR$(24)) in b$ (Zeile 320) ist das Zeichen für die invertierte Darstellung eines Balkens. Durch dieses Zeichen werden die PEN- und PAPER-Farben gegeneinander ausgetauscht, wodurch die Invertierung entsteht. In der Zeile 360 wird erneut die Überschrift gelesen und in Zeile 370 ausgegeben. In der Schleife von 390 bis 470 wird jeweils der Monatsname t$ und der Umsatz x$ gelesen. Mit dem Maßstab mx wird die aktuelle Balkenlänge ix berechnet (Zeile 410). Der Monatsname t$ wird rechts mit genügend vielen Blanks aufgefüllt (Zeile 420). Dann werden von links her t8-Zeichen von t$ in u$ gespeichert (s. Abb. 14). Ist t9

kleiner als ix, so wird in der Zeile 440 u$ rechts mit invertierten Blanks, dem Umsatz x$ und den restlichen Blanks ergänzt. In der Zeile 460 wird u$ ausgegeben.

Das beschriebene Programm kann ohne Änderung für viele andere Anwendungen (z. B. Histogramme in der Statistik) eingesetzt werden. Für die grafische Darstellung des Wahlausganges wären die DATA-Zeilen 120 bis 170 neu einzugeben, indem jeweils der Parteiname gefolgt von der Stimmzahl abgelegt wird.

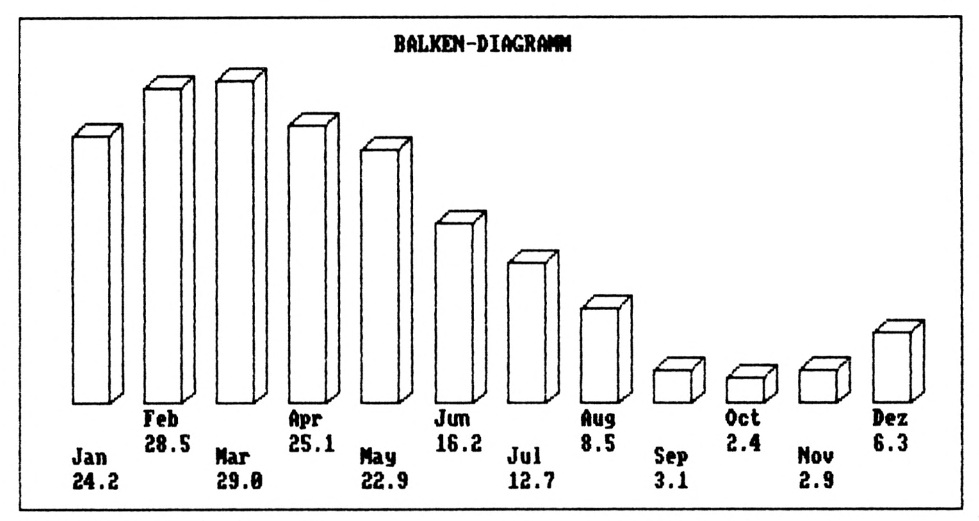


Grafik 49: Histogramm

```
100 REM--- Balkendiagramm ---
110 DATA BALKEN-DIAGRAMM
120 DATA Jan, 24.2, Feb, 28.5
130 DATA Mar, 29.0, Apr, 25.1
140 DATA May, 22.9, Jun, 16.2
150 DATA Jul, 12.7, Aug,  8.5
160 DATA Sep,  3.1, Oct,  2.4
170 DATA Nov,  2.9, Dez,  6.3
180 DATA ee,ee :  REM Kennzeichnung
                  fuer Datenende
190 :
200 i9=79: t8=0: t9=0
205 x8= 0: x9=-3E+30: n=0: READ t$
210 :
215 READ t$,x$
220 WHILE t$<>"ee"
230 : x=VAL(x$):lx=LEN(x$):lt=LEN(t$):n=n+1
240 : IF lt>t8 THEN t8=lt
```

```
250 : IF lx>t9 THEN t9=lx
260 : IF x <x8 THEN x8=x
270 : IF x >x9 THEN x9=x
275 : READ t$,x$
280 WEND
290 :
300 t8 = t8 + 3: t9 = t9 + 5
310 mx = (i9-t9)/(x9-x8)
320 b$ = CHR$(24)
330 FOR k=0 TO i9:b$= b$ + " ":NEXT
340 :
350 MODE 2:BORDER 2:INK 0,1:INK 1,24
355 PAPER 0:PEN 1
360 RESTORE: READ t$
370 PRINT : PRINT SPC(t8); t$: PRINT
380 :
390 FOR k=1 TO n
400 :   READ t$, x$
410 :   x=VAL(x$): lx=LEN(x$): ix=mx*(x-x8)
420 :   t$ = " " + t$ + RIGHT$(b$,i9)
425 :   u$ = LEFT$(t$,t8)
430 :
440 :   IF t9< ix THEN u$ = u$+LEFT$(b$,t9-lx)
        + x$ + RIGHT$(b$,ix-t9) + CHR$(24)
450 :   IF t9>=ix THEN u$ = u$+LEFT$(b$,ix)
        + CHR$(24) + " " + x$
460 :   PRINT u$
470 NEXT k
```

Wir wollen nun mit der hochauflösenden Grafik senkrechte 3D-Balkendiagramme zeichnen. Zum Zeichnen der dreidimensionalen Balken benutzen wir die MOVE- und DRAW-Befehle. Zunächst wird die Vorderseite des Balkens, die einem Rechteck entspricht, gezeichnet (Zeile 840 bzw. 850). Danach werden durch die Zeilen 860 bis 900 noch einige Linien hinzugefügt, so daß ein dreidimensionaler Balken entsteht. In der Zeile 510 ist die Überschrift des Diagrammes hinterlegt. Die Zeilen 520 bis 570 enthalten jeweils das Kennzeichen (Monatsnamen) für den Balken und die Balkenhöhe (Umsatz). Die Endekennung (ee,ee) der Daten ist in der Zeile 580 hinterlegt. Indem wir die Monatsnamen durch Parteien-Kennzeichen und den Umsatz durch die Sitzzahl ersetzen, können wir dieses Programm auch zur grafischen Darstellung von Wahlergebnissen verwenden. In der Schleife 620 bis 690 wird die Anzahl n der eingegebenen Daten hochgezählt (Zeile 625) und der maximale Umsatz (y9) ermittelt. Durch die Vorbesetzung y8=0 (Zeile 600,630) wird für positive Umsatzzahlen der Wert y8=0 nicht verändert. Die Werte y8,y9 werden in der Zeile 720



Grafik 50: 3D-Balkendiagramm

zur Berechnung des Maßstabsfaktors my verwendet. Die gezeichneten Balkenhöhen liegen zwischen den y-Grenzen j8=85.5 und j9=345.5 (Zeile 605).

In der Zeile 740 wird der Datenzeiger durch RESTORE zurückgesetzt, die Diagrammüberschrift gelesen und durch Zeile 760 in die Grafik geschrieben. Z. B. wird durch

```
LOCATE x, y: PRINT A$
```

der in A$ gespeicherte Text ab der Spalte x und der Zeile y auf den Bildschirm geschrieben. Um den Text auf Grafik-Cursor-Positionen auszugeben und um das Diagramm zu beschriften, muß zunächst das TAG-Kommando gegeben werden (Zeile 785). Danach wird durch

```
MOVE x, y: PRINT A$
```

der Cursor an der Stelle mit den Bildschirmkoordinaten(!) (x,y) positioniert und durch den PRINT-Befehl der in A$ gespeicherte Text ab (x,y) auf den Bildschirm geschrieben (Zeilen 920,930). Durch TAGOFF (Zeile 950) wird das TAG-Kommando aufgehoben. In der Schleife 790 bis 940 werden die n senkrechten Balken, wie in der Grafik 50 dargestellt, perspektivisch gezeichnet.

```
500 REM--- Balkendiagramm ---
510 DATA BALKEN-DIAGRAMM
520 DATA Jan, 24.2, Feb, 28.5
530 DATA Mar, 29.0, Apr, 25.1
540 DATA May, 22.9, Jun, 16.2
550 DATA Jul, 12.7, Aug,  8.5
560 DATA Sep,  3.1, Oct,  2.4
```

```
570 DATA Nov,  2.9, Dez,  6.3
580 DATA ee,ee : REM Kennzeichnung
                 fuer Datenende
590 :
600 y8 =  0  : y9 =-3E+30: n=0
605 j8 = 85.5: j9 = 345.5: READ t$
610 :
615 READ x$, y$
620 WHILE x$<>"ee"
625 :  y = VAL(y$): n = n + 1
630 :  IF y<y8 THEN y8=y
640 :  IF y>y9 THEN y9=y
650 :  READ x$, y$
690 WEND
695 :
700 dx = 639/(n+1): dl = dx/5
720 my = (j9-j8)/(y9-y8)
730 :
740 RESTORE: READ t$
750 MODE 2:BORDER 1:INK 0,3:INK 1,26
760 LOCATE 33, 2: PRINT t$
770 MOVE  0,0: DRAW 639,0,1: DRAW 639,399,1:
    DRAW 0,399,1: DRAW 0,0,1
775 jl = j8 - my*y8
780 :
785 TAG
790 FOR k=1 TO n
800 :  READ x$, y$
810 :  y = VAL(y$):  h = 30*(k MOD 2)
820 :  i% = (k-0.25)*dx: il = i% + 0.5*dx
830 :  j% = j8 + my*(y-y8)
835 :
840 : IF jl>=j% THEN MOVE i%,j%: DRAW il,j%,1:
      DRAW il,jl,1:DRAW i%,jl,1: DRAW i%,j%,1
845 :
850 : IF jl< j% THEN MOVE i%,jl: DRAW il,jl,1:
      DRAW il,j%,1:DRAW i%,j%,1: DRAW i%,jl,1
855 :
860 : MOVE   i% ,  j%   : DRAW i%+dl,j%+dl,1
870 : DRAW il+dl,j%+dl,1: DRAW   il ,  j% ,1
890 : MOVE il+dl,j%+dl  : DRAW il+dl,jl+dl,1
900 : DRAW   il ,  jl ,1
```

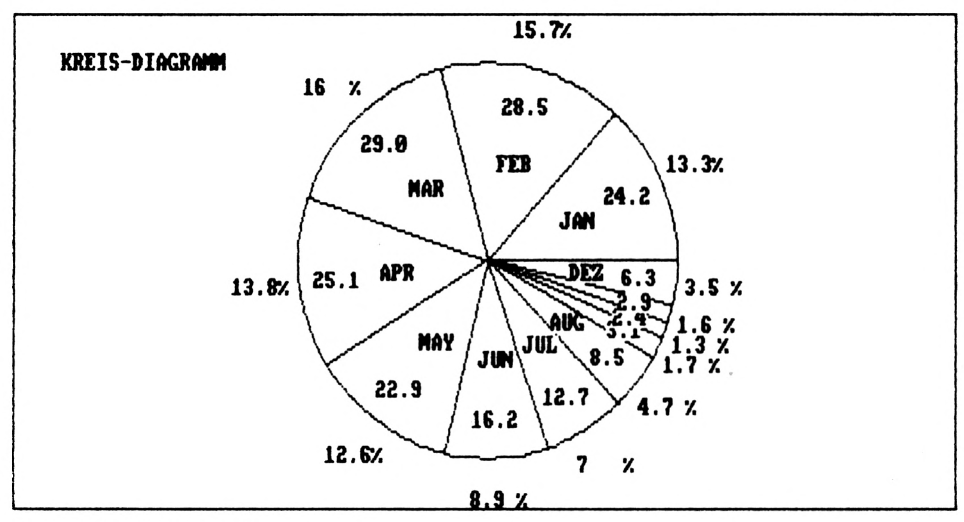
```
905 :
920 :   MOVE i%,j8-h-5 : PRINT x$;
930 :   MOVE i%,j8-h-25: PRINT y$;
940 NEXT k
950 TAGOFF
960 :
970 a$=INKEY$: IF a$="" THEN 970
```

11.2 Kreisdiagramme

Zur Veranschaulichung von Verteilungsproblemen werden häufig Kreisdiagramme benutzt. Beispiele sind die Verteilung
- der Parlamentssitze auf die Parteien
- des Geschäftsumsatzes auf die Abrechnungsperioden
- der Klassenarbeiten auf die Noten
- bei Prognosedarstellungen
- des Etats einer Kommune auf die Ressorts.

Wir wollen ein universell anwendbares Programm zum Zeichnen eines Kreisdiagrammes entwickeln. Hierzu betrachten wir als Beispiel den monatlichen Umsatz (in 1000 DM) eines Unternehmens. Die Monatsbezeichnung und der dazugehörige Umsatz werden in den DATA-Zeilen 110 bis 140 agelegt. Als Endekennung für die Daten wird ee (Zeile 150) verwendet. Der Kreisradius in x- bzw. y-Richtung ist mit a bzw. b bezeichnet (Zeile 170). Der Kreismittelpunkt hat die Koordinaten x0,y0. In der Schleife 220,230 wird der Gesamtumsatz in der Variablen sx aufsummiert. Das Hochzählen der Abrechnungsperioden (hier die Anzahl der Monate) erfolgt in der Variablen n. Wird das Endekriterium gelesen, so verzweigt das Programm zur Zeile 250. In unserem Falle ist dann n=12. Nach RESTORE (Zeile 260) wird die Überschrift erneut gelesen und in der Zeile 270 ausgegeben. Durch die Zeilen 280 bis 350 wird ein Kreis um den Mittelpunkt (x0,y0) gezeichnet. Die Änderung des a/b-Verhältnisses (Zeile 170) gestattet auch eine Kreisdarstellung auf dem Drucker. Der gelesene monatliche Umsatz x$ wird durch den Gesamtumsatz sx in der nachfolgenden Schleife (Zeilen 380 bis 490) dividiert. Dieser Wert wird mit pp=2∗π multipliziert und ergibt den Zentriwinkel w des Kreissektors im Bogenmaß (Zeile 390). Der bisher überstrichene Zentriwinkel w des Kreissektors wird in der Variablen sw aufsummiert (Zeile 400).

Die Zeilen 420 und 430 berechnen den aktuellen Kreispunkt (x2,y2), die Zeile 440 veranlaßt das Zeichnen eines Radius vom Kreismittelpunkt (x0,y0) zum Kreispunkt(x2,y2). Dieser Radius bildet den entsprechenden Kreissektor (s. Grafik 51). Um die Grafik zu beschriften, wird in Zeile 370 mit Hilfe des TAG-Befehls die Ausgabe von Text auf Grafikcursorpositionen ermöglicht. Danach kann Text mit dem PRINT-Befehl an jeder beliebigen Position des Bildschirmes ausgegeben werden. Durch TAGOFF (Zeile 500) wird das TAG-Kommando aufgehoben. Zur Beschriftung der Grafik wird der halbe Winkel w/2 verwendet (Zeilen 420,430). Da die Beschriftung bei der aktuellen Cursor-Position beginnend nach



Grafik 51: Kreisdiagramm

rechts hin ausgegeben wird, muß die Textausgabe abhängig von der Textlänge nach links versetzt werden. Hierzu dienen die Zeilen 450,460,470 und 480. Die Linksverschiebung des Textanfangs hängt von den Textlängen lp,lx bzw. lt ab. Die Berechnung der prozentualen Verteilung p$ des Umsatzes (x bzw. w) geschieht in der Zeile 410. Hierbei wird eine Rundung auf eine Kommastelle durchgeführt.

Dieses Programm zum Zeichnen eines Kreisdiagrammes kann für viele Anwendungen ohne Änderung verwendet werden. Lediglich die Zeilen 110 bis 140 sind durch andere DATA-Zeilen zu ersetzen. Die Anzahl der Kreissektoren ist variabel und wird durch das Lesen bis zum Endekriterium automatisch ermittelt. Die Überschrift für das Kreisdiagramm ist in der DATA-Zeile 100 zu hinterlegen.

```
100 DATA KREIS-DIAGRAMM
110 DATA JAN, 24.2, FEB, 28.5, MAR, 29.0
120 DATA APR, 25.1, MAY, 22.9, JUN, 16.2
130 DATA JUL, 12.7, AUG,  8.5, SEP,  3.1
140 DATA OCT,  2.4, NOV,  2.9, DEZ,  6.3
150 DATA ee,ee : REM Datenende
160 :
170 b = 160: a = 1.1*b: x0 = 320: y0 = 200
180 x9 = -3E+30: pp = 2*PI: sx = 0: n = 0
190 READ t$
195 :
```

```
200 READ t$,x$
210 WHILE t$<>"ee"
220 :   sx = sx + VAL(x$): n = n +1
230 :   READ t$,x$
240 WEND
245 :
250 MODE 2:BORDER 2:INK 0,3:INK 1,26
255 :
260 RESTORE: READ t$: sw = 0
270 LOCATE 5,3: PRINT t$
275 :
280 ORIGIN x0,y0
290 DEG
300 FOR t=0 TO 360 STEP 5
310 :   x2 = a*COS(t): y2 = b*SIN(t)
320 :   IF t>0 THEN MOVE x1,y1: DRAW x2,y2,1
330 :   x1 = x2: y1 = y2
340 NEXT t
350 RAD
355 :
360 ORIGIN 0,0
370 TAG
380 FOR k=1 TO n: READ t$,x$
390 :   w = pp/sx*VAL(x$)
400 :   sw = sw + w
410 :   p$ = MID$(STR$(INT(1000*w/pp+0.5)/10)
        +"    ",2,4)+"%"
420 :   x2 = x0 + a*COS(sw): x3 = a*COS(sw-w/2)
430 :   y2 = y0 + b*SIN(sw): y3 = b*SIN(sw-w/2)
440 :   MOVE x0,y0: DRAW x2,y2,1
450 :   lt = LEN(t$): lp = LEN(p$): lx = LEN(x$)
460 :   MOVE x0+1.2*x3-4*lp, y0+1.2*y3+6:
        PRINT p$;
470 :   MOVE x0+0.8*x3-4*lx, y0+0.8*y3+6:
        PRINT x$;
480 :   IF w>0.2 THEN
        MOVE x0+0.5*x3-3*lt, y0+0.5*y3+6:
                                        PRINT t$;
490 NEXT k
500 TAGOFF
510 :
520 a$=INKEY$: IF a$="" THEN 520
```

12. Zufallsgrafiken

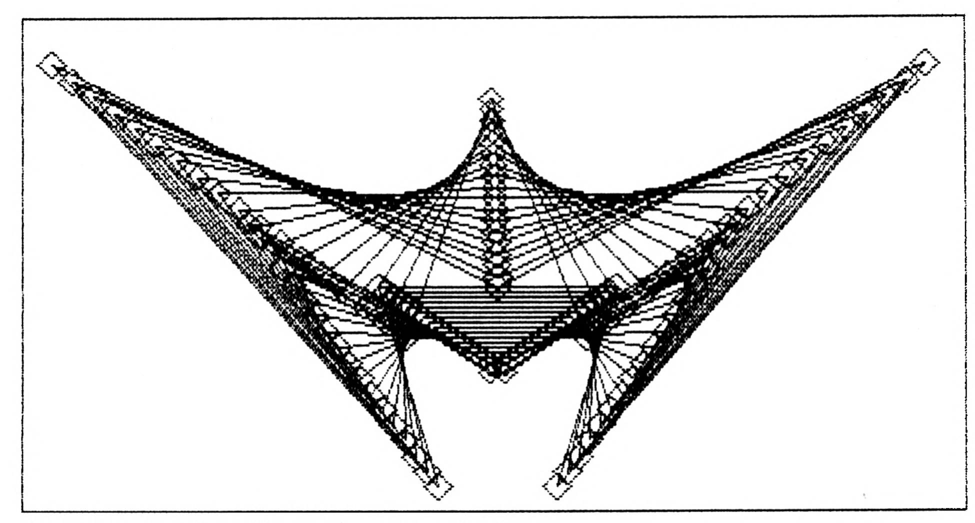
Zufallseinflüsse für Computergrafiken können in BASIC mit Hilfe der RND(1)-Funktion realisiert werden. Diese Funktion erzeugt einen zwischen 0 und 1 liegenden Zufallswert. In den Kapiteln 2.1, 2.2, 2.8 haben wir bereits solche Zufallsgrafiken erzeugt. Hier soll nun eine zufällige Folge von n-Punkten P1(u1,v1),P2(u2,v2),. . .,Pn(un,vn) aufgebaut werden. Durch Verbinden der Punkte können vielfältige Grafiken erzeugt werden. Jeder Programmablauf liefert neue Bilder, so daß der Beobachter entscheiden kann, ob das generierte Bild seinen ästhetischen Ansprüchen genügt.

Die in den Programmzeilen 240 und 250 erzeugten Zufallskoordinaten u(k),v(k) liegen zwischen den Werten

```
60 < u(k) < 580
40 < v(k) < 360.
```

Die Zeile 260 bewirkt das Entfernen von Punkten aus der Bildmitte. Die Anfangs- und Endpunkte der Zufallsfolge werden in Zeile 290 und 300 so verknüpft, daß später geschlossene Linienzüge entstehen.

In den Zeilen 640,650 wird entsprechend dem t-Parameter ein „Mittelwert" von 3 aufeinanderfolgenden Zufallspunkten gebildet. Diese neuen Punkte werden durch Geraden verbunden. Somit ergibt sich ein „gemitteltes" Zufalls-n-Eck (Zeilen 630 bis 710). Dabei ergeben sich durch den Mittelungsparameter t1 in Zeile 340 Zufallsgrafiken wie z. B. Grafik 52.



Grafik 52: „Zufallsgrafik" mit vorgegebenen Punkten

Verwenden wir statt der Zeilen 240 bis 260 lediglich

```
260 READ u(k), v(k)
```

und ersetzen außerdem 220 durch

```
220 READ n
```

so ergibt sich mit den 7 festen Punkten gemäß den folgenden DATA-Zeilen

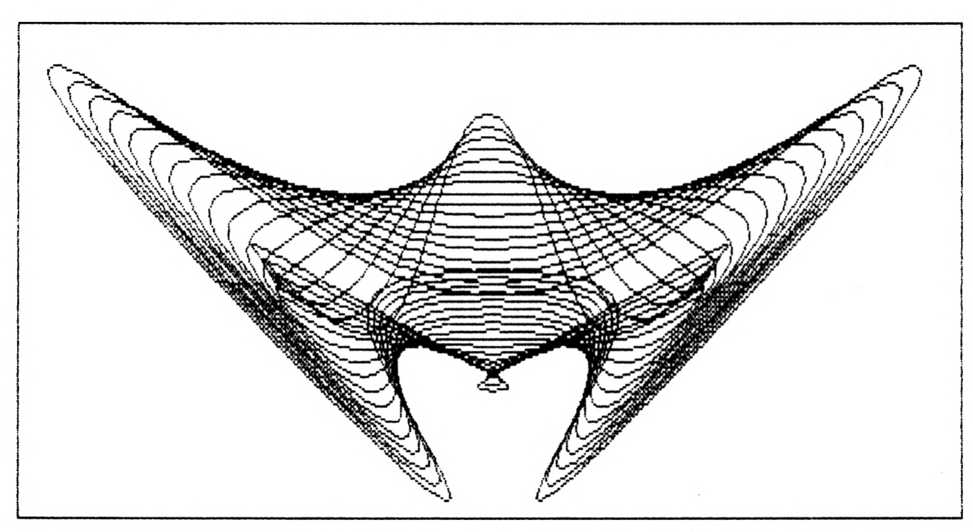
```
190 DATA 7,240,180,280,20,20,360,320
195 DATA 180,610,360,360,20,400,180
```

die Grafik 52. Durch die Eingabe der festen Punkte ist die Programmausgabe eindeutig und somit für den Leser überprüfbar.

```
200 REM--- Zufallsgrafiken mit Geraden ---
210 DIM u(40), v(40)
220 n = 12
230 FOR k=1 TO n
240 : u(k) = 60 + 520*RND(1)
250 : v(k) = 40 + 320*RND(1)
260 : IF ABS(v(k)-200)<80 OR
      ABS(u(k)-320)<120 THEN 240
270 NEXT k
280 :
290 u(0)=u(n) : u(n+1)=u(1) : u(n+2)=u(2)
300 v(0)=v(n) : v(n+1)=v(1) : v(n+2)=v(2)
310 :
320 MODE 2:BORDER 4:INK 0,3:INK 1,26
330 :
340 FOR t1=0 TO 0.9 STEP 0.05
350 :
630 : FOR k=1 TO n+1
640 :   x2 = u(k) + t1*((u(k-1)+u(k+1))/2-u(k))
650 :   y2 = v(k) + t1*((v(k-1)+v(k+1))/2-v(k))
660 :   IF k>1 THEN MOVE x1, y1: DRAW x2, y2, 1
670 :   MOVE x2-(10-2*t1),y2
675 :   DRAW x2,y2+(10-2*t1), 1
680 :   DRAW x2+(10-2*t1),y2, 1
685 :   DRAW x2,y2-(10-2*t1), 1
690 :   DRAW x2-(10-2*t1),y2, 1
700 :   x1 = x2 : y1 = y2
710 : NEXT k
720 NEXT t1
730 :
740 a$=INKEY$: IF a$="" THEN 740
```

Bei dem obigen Programm wurden die Zufallspunkte durch Geraden verbunden. Jeder Eckpunkt wird durch ein kleines Viereck (Zeilen 670 bis 690) markiert. Sollen die Zufallspunkte durch **glatte Kurven** (stetige 1. und 2. Ableitung) verbunden werden, so kann z. B. das im Kapitel14.3 beschriebene Verfahren angewendet werden (Zeilen 1485 bis 1580).

In den Zeilen 1630–1700 werden die Koordinaten von 3 aufeinanderfolgenden Punkten entsprechend dem t1-Wert gemittelt und ergeben dadurch neue Punkte. Indem wir die Schrittweite in der Zeile 1470 ändern, erhalten wir einen anderen Linienabstand. Verwenden wir auch hier die 7 festen Punkte des vorhergehenden Programmes, so liefert das Programm die Grafik 53.



Grafik 53: Zufallsgrafik mit interpolierenden Linienzügen bei vorgegebenen Punkten

```
1200 REM- Zufallsgrafiken mit stetigen Kurven -
1210 DIM u(40), v(40), x(40), y(40)
1220 n=12
1230 FOR k=1 TO n
1240 :   u(k) = 60 + 520*RND(1): x(k) = u(k)
1250 :   v(k) = 40 + 320*RND(1): y(k) = v(k)
1260 :   IF ABS(v(k)-200)<80 OR
         ABS(u(k)-320)<120 THEN 1240
1270 NEXT k
1280 :
1290 u(0) = u(n): u(n+1) = u(1): u(n+2) = u(2)
1300 v(0) = v(n): v(n+1) = v(1): v(n+2) = v(2)
1310 :
1320 MODE 2:BORDER 2:INK 0,1:INK 1,26
1330 MOVE 0,0: DRAW 0,399,1: DRAW 639,399,1
```

```
1335 DRAW 639,0,1: DRAW 0,0,1
1337 :
1340 FOR t1=0.05 TO 0.9 STEP 0.05
1350 :
1400 : REM--- Verbinden der Zufallspunkte ---
1410 : REM---  durch stetigen Linienzug   ---
1430 : x(0)  = x(n): x(n+1)  = x(1): x(n+2)  = x(2)
1440 : y(0)  = y(n): y(n+1)  = y(1): y(n+2)  = y(2)
1450 :
1460 : FOR k=0 TO n-1
1470 :  FOR t=0 TO 1.01 STEP 0.1
1485 :   xa =  (x(k)  + x(k+2))/2  - x(k+1)
1490 :   ya =  (y(k)  + y(k+2))/2  - y(k+1)
1495 :   xb =  (x(k+1)  + x(k+3))/2  - x(k+2)
1500 :   yb =  (y(k+1)  + y(k+3))/2  - y(k+2)
1510 :   xa = x(k)  +  (t+1)  *  (x(k+1)-x(k)+xa*t)
1520 :   ya = y(k)  +  (t+1)  *  (y(k+1)-y(k)+ya*t)
1530 :   xb = x(k+1)  +   t   *   (x(k+2)  - x(k+1)
         + xb*(t-1))
1540 :   yb = y(k+1)  +   t   *   (y(k+2)  - y(k+1)
         + yb*(t-1))
1545 :   t0  = t*t*(3-2*t): REM oder t0=t
1550 :   x2  = xa +  (xb-xa)*t0
1560 :   y2  = ya +  (yb-ya)*t0
1570 :   IF t>0 THEN MOVE x1,y1: DRAW x2,y2, 1
1580 :   x1  = x2: y1  = y2
1590 :  NEXT t
1600 : NEXT k
1620 :
1630 : FOR k=1 TO n
1640 :   x(k)  = u(k)  + t1*((u(k-1)
         + u(k+1))/2  - u(k) )
1650 :   y(k)  = v(k)  + t1*((v(k-1)
         + v(k+1))/2  - v(k) )
1700 : NEXT k
1710 NEXT t1
1720 :
1730 a$=INKEY$ : IF a$="" THEN 1730
```

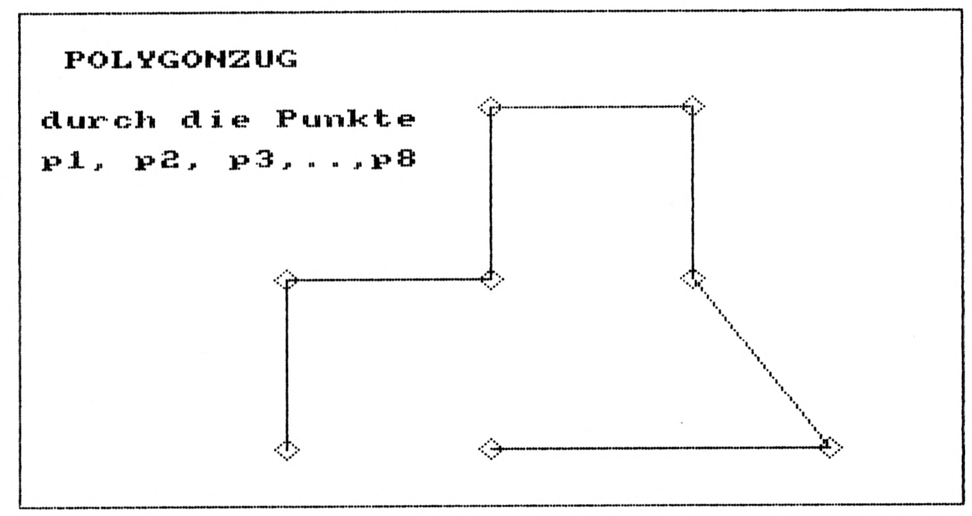
13. Ebener Polygonzug

Bei vielen naturwissenschaftlichen Anwendungen ist eine gewisse Anzahl von Punkten in der x,y-Ebene vorgegeben. Z. B. liefert die Durchführung von Experimenten aufeinanderfolgende Punkte in einem x,y-System, oder die Eckpunkte eines Grundstückes sind durch x,y-Koordinaten festgelegt. Im einfachsten Fall sollen diese Punkte durch Geraden verbunden werden (Grafik 54).

Die Punkte P1,P2,P3,. . .,P8 haben die Koordinaten x1,y1,x2,y2,. . .,x8,y8. Diese Koordinaten sind als DATA-Werte (Zeile 40,50) abgelegt. Der erste DATA-Wert (hier 8) gibt an, daß es sich um 8 Punkte handelt. Diese Zahl 8 wird in den n-Speicher (Zeile 120) gelesen und die Koordinaten der Punkte in die x(k), y(k)-Feldelemente (Zeile 140). Mit den kleinsten (x8) und größten (x9) x-Achsenwerten, sowie den kleinsten (y8) und größten (y9) y-Werten (Zeile 200) werden die Maßstäbe mx,my berechnet (Zeile 300). Die Umrechnung der Koordinaten x(k),y(k) des k-ten Punktes in den Bildschirmpunkt (i2,j2) (Zeile 350,360) erfolgt in der k-Schleife (Zeilen 340—410). Um jeden Punkt wird ein Markierungsquadrat gezeichnet (Zeile 380,390), und mit Hilfe der DRAW-Anweisung (Zeile 370) werden Linien vom 1. zum 2. Punkt, vom 2. zum 3. Punkt usw. gezeichnet. Durch die Punkte P1,P2,. . .,P8 erhalten wir einen Polygonzug. Durch die Zeilen 430,440,450 werden die angegebenen Texte in die Grafik geschrieben (s. Grafik 54).

```
10 REM--- Polygonzug durch p1,p2,p3,...,pn ---
15 :
20 REM  n, x1,y1, x2,y2, ..............., xn,yn
40 DATA 8, -3,-3, -3,0, 0,0, 0,3
50 DATA 3,3, 3,0, 5,-3, 0,-3
90 :
100 :
110 DIM x(100), y(100)
115 :
120 READ n
130 FOR k=1 TO n
140 :  READ x(k),y(k)
150 NEXT k
170 :
200 x8 = -7 : x9 =  7: y8 = -4: y9 =  3
210 i8 = 0.5: i9 =639.5: j8 = 0.5: j9 =319.5
220 :
300 mx = (i9-i8)/(x9-x8): my = (j9-j8)/(y9-y8)
310 :
320 MODE 1:BORDER 10:INK 0,1:INK 1,26
```

```
330 :
340 FOR k=1 TO n
350 : i2 = i8 + mx * (x(k) - x8)
360 : j2 = j8 + my * (y(k) - y8)
370 : IF k>1 THEN MOVE i1,j1: DRAW i2,j2, 1
380 : MOVE i2-8,j2: DRAW i2,j2+8, 1
390 : DRAW i2+8,j2, 1: DRAW i2,j2-8, 1
395 : DRAW i2-8,j2, 1
400 : i1 = i2: j1 = j2
410 NEXT k
420 :
430 LOCATE 3,3: PRINT "POLYGONZUG"
440 LOCATE 2,6: PRINT "durch die Punkte"
450 LOCATE 2,8: PRINT "p1, p2, p3,..,p8"
490 :
500 a$=INKEY$: IF a$="" THEN 500
```



Grafik 54: Ebener Polygonzug

Entnehmen wir einem Atlas koordinatenmäßig die Punkte der Bundesrepublik und legen diese in DATA-Zeilen ab, so wird nach dem Programmstart die Grafik 55 gezeichnet.

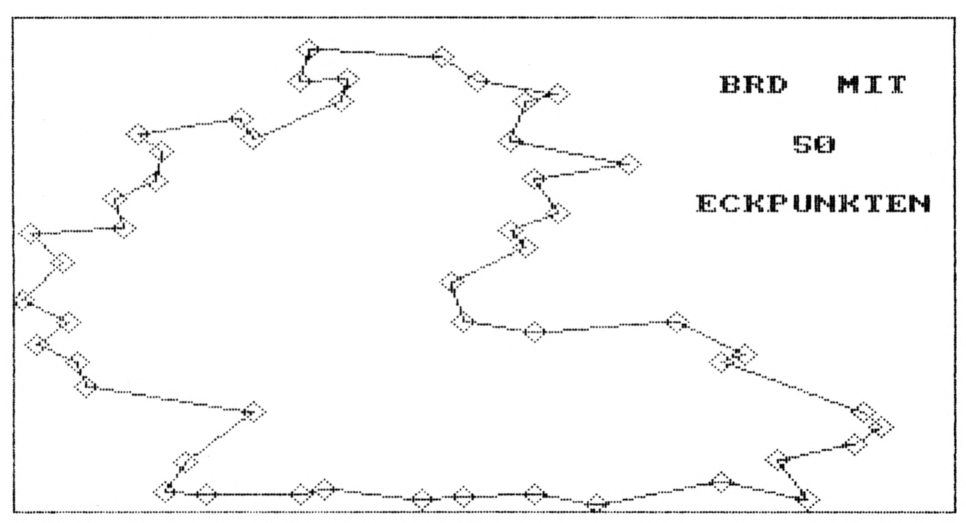
Damit kein Abzählen der Anzahl von Punkten erforderlich ist, werden die Daten (s. Zeile 110) mit dem Endekennzeichen 9E+09 abgeschlossen. Wird das Datenende gelesen (Zeile 140), so wird infolge der Zeile 150 die Lese-Schleife verlassen. Die Variable n enthält dann die Anzahl der gelesenen Punkte.

Um eine geschlossene Figur zu erhalten, wird vor dem 1. Eckpunkt der n. Eckpunkt abgelegt (Zeile 180). Dadurch wird eine Linie vom letzten zum ersten Eckpunkt gezeichnet und

der Polygonzug geschlossen (Zeile 380). Durch die Zeilen 430,440,450 wird die Grafik 55beschriftet.

```
10 REM- geschlossener Polygonzug fuer die BRD -
15 :
20 REM   x1  , y1  , x2  , y2  ,.........,  xn  , yn
25 DATA  4.0, 1.0, 6.0, 1.0, 6.5, 1.3, 8.6, 0.8
30 DATA  9.5, 0.9,11.0, 1.0,12.3, 0.5,15.0, 1.8
35 DATA 16.8, 0.8,16.2, 3.2,17.8, 4.1,18.4, 5.1
40 DATA 18.0, 6.0,15.0, 9.0,15.5, 9.5,14.0,11.5
45 DATA 11.0,10.8, 9.5,11.5, 9.2,13.8,10.8,16.0
50 DATA 10.5,17.0,11.5,18.0,11.0,20.2,13.0,21.0
55 DATA 10.5,22.4,10.8,24.8,11.5,25.3, 9.8,26.0
60 DATA  9.0,27.5, 6.2,28.0, 6.0,26.0, 7.0,26.2
65 DATA  6.9,24.8, 5.0,22.5, 4.7,23.8, 2.5,22.9
70 DATA  3.0,21.8, 2.9,20.0, 2.0,19.0, 2.2,17.1
75 DATA  0.2,16.8, 0.9,15.1, 0.0,12.8, 1.0,11.4
80 DATA  0.3,10.1, 1.2, 9.0, 1.4, 7.5, 5.0, 6.0
85 DATA  3.5, 3.0, 3.1,1.2
90 :
95 REM   Ende - Kennung
110 DATA  9E+09, 9E+09
115 :
120 DIM x(100), y(100)
125 :
130 FOR k=1 TO 1000
140 :  READ x(k),y(k)
150 :  IF x(k)=9E+09 AND y(k)=9E+09
       THEN n=k-1: k=9999
160 NEXT k
170 :
180 x(0) = x(n): y(0) = y(n): REM geschlossen
210 :
220 x8 =-0.2: x9 = 20  : y8 = 0  : y9 = 30
230 i8 = 0.5: i9 =639.5: j8 = 0.5: j9 =399.5
290 :
300 mx = (i9-i8)/(x9-x8): my = (j9-j8)/(y9-y8)
310 :
320 MODE 1:BORDER 1:INK 0,2:INK 1,26
330 :
340 FOR k=0 TO n
350 :  i2 = i8 + mx * (x(k) - x8)
```

```
360 :   j2 = j8 + my * (y(k) - y8)
380 :   IF k>0 THEN MOVE i1,j1: DRAW i2,j2, 1
390 :   MOVE i2-8,j2: DRAW i2,j2+8, 1
395 :   DRAW i2+8,j2, 1: DRAW i2,j2-8, 1
400 :   DRAW i2-8,j2, 1
410 :   i1 = i2: j1 = j2
420 NEXT k
425 :
430 LOCATE 31, 4: PRINT "BRD  MIT"
440 LOCATE 33, 7: PRINT STR$(n)
450 LOCATE 30,10: PRINT "ECKPUNKTEN"
460 :
500 a$=INKEY$: IF a$="" THEN 500
```



Grafik 55: BRD als geschlossener Polygonzug

In vielen Fällen benötigt man auch den Wert der Fläche eines geschlossenen Polygonzuges. In dem Buch „Bachmann, Haake: Matrizenrechnung für Ingenieure, Springer-Verlag" ist das Verfahren erklärt. Hier soll lediglich das Programm angegeben werden. Zur Berechnung der n-Eck-Fläche ist die Anzahl n der Eckpunkte (Zeile 45) einzugeben. In der Zeile 50 sind die Koordinaten der Eckpunkte abgelegt. Hierbei ist darauf zu achten, daß die Reihenfolge der Eckpunkte so gewählt wird, daß die gesuchte Fläche immer links vom Umlaufweg liegt. Wenn wir dies beachten, so können auch Hohlräume umlaufen werden. Die 6 Eckpunkte des folgenden Programmes ergeben einen Flächenwert von 18.

```
10 REM--- Flaechenberechnung eines ge- ---
20 REM---  schlossenen Polygonzuges    ---
30 REM---  Umlaufrichtung beachten !!  ---
35 :
40 REM Anzahl n der Punkte
45 DATA   6
50 DATA -1,-1,  5,-1,  5,3,  2,3,  2,1,  -1,1
90 :
115 :
120 READ n
130 DIM x(n), y(n)
135 :
140 FOR k=1 TO n
150 :  READ x(k),y(k)
160 NEXT k
210 :
220 x(0) = x(n): y(0) = y(n)
230 :
240 FOR k=1 TO n
250 :  a = a + x(k-1)*y(k) - x(k)*y(k-1)
260 NEXT k
270 :
280 a = a/2
290 PRINT " Flaeche  = ";a
```

14. Approximationen

Bei vielen Anwendungen sind Punkte in der x,y-Ebene vorgegeben, und ein Kurvenzug wird gesucht, der diese Punkte beschreibt. Liegen sehr viele Meßpunkte vor, so ist oft ein „mittlerer“ Kurvenzug gesucht, der zwar nicht durch jeden Meßpunkt geht, aber die „Punktewolke“ im Sinne der kleinsten Fehlerquadrate beschreibt. Wir sprechen dann von einer Regressionskurve.

Dagegen führt ein interpolierender Kurvenzug durch jeden vorgegebenen Punkt. Dieser interpolierende Kurvenzug approximiert die unbekannte Funktion, von der lediglich die Stützstellen bekannt sind.

14.1 Interpolation nach Lagrange

In der Ebene seien die drei Punkte P0(x0,y0),P1(x1,y1) und P2(x2,y2) gegeben (s. Abb. 15). Diese 3 Punkte sollen durch einen parabolischen Linienzug verbunden werden. Mit Hilfe der LAGRANGESCHEN Interpolationsformel erhalten wir die Gleichung

```
x(t) = x0*(t-1)*t/2 + x1*(1-t)*(1+t) + x2*(t+1)*t/2
y(t) = y0*(t-1)*t/2 + y1*(1-t)*(1+t) + y2*(t+1)*t/2
```

in Parameterdarstellung. Setzen wir z. B. t=−1 ein, so ergibt sich x(−1)=x0,y(−1)=y0. Für t=0 ergibt sich x(0)=x1,y(0)=y1 und für t=1 ergibt sich nach dem Einsetzen in die obigen Formeln x(1)=x2,y(1)=y2. Wenn t von −1 bis +1 läuft, so ergeben sich Zwischenpunkte (x(t),y(t)), die auf einem parabolischen Linienzug durch die vorgegebenen Punkte P0,P1,P2 liegen (Abb. 15). Diese Parameterdarstellung hat gegenüber der gewöhnlichen kartesischen Parabeldarstellung y=a+b*x+c*x*x den Vorteil, daß z. B. auch „U“ oder „C“-förmige Linienzüge eindeutig dargestellt werden können. Das folgende Programm liefert (abgesehen von der Form) die Wertetabelle:

t	−1.0	−.8	−.6	−.4	−.2	0.0	0.2	0.4	0.6	0.8	1.0
x	2.0	4.4	6.4	8.0	9.2	10.0	10.4	10.4	10.0	9.2	8.0
y	−2.0	−.48	0.88	2.08	3.12	4.0	4.72	5.28	5.68	5.92	6.0

```
100 REM--- Wertetabelle ---
105 REM   x0,y0,   x1,y1,   x2,y2
110 DATA  2,-2,   10, 4,     8, 6
120 :
130 READ x0,y0,   x1,y1,   x2,y2
135 :
140 FOR t=-1 TO 1 STEP 0.2
```

```
150 :   x = x0*(t-1)*t/2 + x1*(1-t)*(1+t)
        + x2*(t+1)*t/2
160 :   y = y0*(t-1)*t/2 + y1*(1-t)*(1+t)
        + y2*(t+1)*t/2
170 :   PRINT  t, x, y
180 NEXT t
```

In einem x,y-System aufgetragen, erhalten wir die folgende Abbildung:

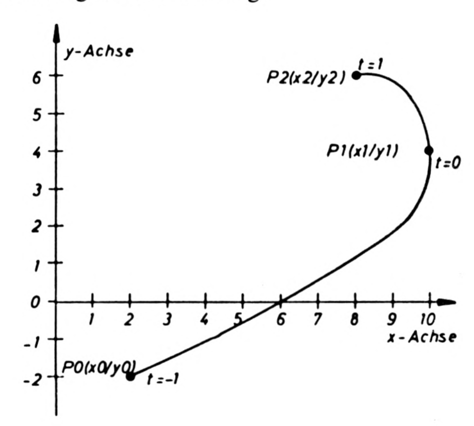


Abb. 15 Parabolische Interpolation durch die gegebenen Punkte P0,P1,P2

14.2 Fortlaufende parabolische Interpolation

Bei technischen Anwendungen tritt häufig das Problem auf, daß die in der x,y-Ebene liegenden Punkte P1,P2,P2,. . .,Pn fortlaufend durch einen Kurvenzug zu verbinden sind. Der Kurvenzug soll glatt und ohne spitze Ecken durch die vorgegebenen Punkte gehen. Ein interpolierendes Polynom (n−1)-ten Grades weist in vielen Fällen eine große Welligkeit auf. Deshalb wird die folgende Methode bevorzugt.

Zunächst betrachten wir die Abb. 16. Die Koordinaten x0,y0,x1,y1,x2,y2,x3,y3 der Punkte P0,P1,P2 und P3 seien bekannt. Die Parabel durch P0,P1,P2 kann durch

```
xa = x0*(t-1)*t/2 + x1*(1-t)*(1+t) + x2*(t+1)*t/2
ya = y0*(t-1)*t/2 + y1*(1-t)*(1+t) + y2*(t+1)*t/2
```

dargestellt werden (s. 8. u. 14.1).

Für t=−1,−0.8,−0.6,. . .,+0.8,+1 ergeben sich Punkte Pa(xa,ya), die auf der durchgezogenen Linie von P0 nach P1 nach P2 liegen. Diese Parabel nennen wir die a-Parabel.

In gleicher Weise können wir eine Parabel durch P1,P2,P3 konstruieren. In der Abb. 16 ist diese Parabel strichpunktiert. Diese strichpunktierte Parabel nennen wir die b-Parabel, die durch

```
xb = x1*(t-1)*(t-2)/2 + x2*(2-t)*t + x3*(t-1)*t/2
yb = y1*(t-1)*(t-2)/2 + y2*(2-t)*t + y3*(t-1)*t/2
```

gegeben ist.

Für t=0 ist xb=x1,yb=y1. Für t=1 ist xb=x2,yb=y2. Für t=2 ist xb=x3,yb=y3. Diese Parabel geht offensichtlich durch die Punkte P1,P1,P3. Die Abb. 16 macht deutlich, daß die a- und b-Parabelstücke i. allg. verschieden sind.

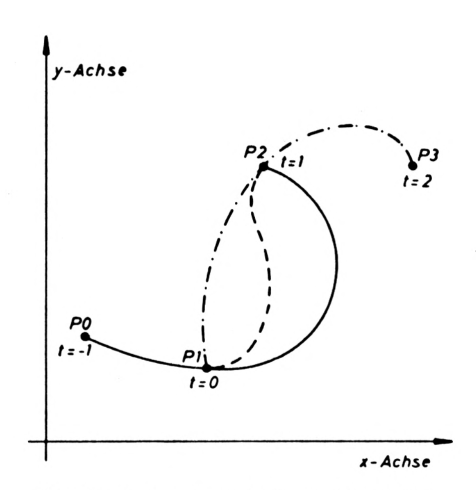


Abb. 16: Interpolierende Kurve (gestrichelt)

Wir wollen im folgenden lediglich den Kurvenverlauf von P1 nach P2 betrachten. Der Parameter t läuft hier von 0 bis 1. Für diesen Bereich 0<=t<=1 wählen wir eine gemittelte Kurve, die in der Abb. 16 gestrichelt gezeichnet ist. Für „kleine" t-Werte (z. B. 0<=t<=0.1) soll diese gestrichelte Kurve mehr der a-Parabel entsprechen. Für „große" t-Werte (z. B. 0.9<=t<=1) soll diese gesuchte Kurve mehr der b-Parabel entsprechen. Wenn xa,xb und ya,yb für einen Punkt zwischen P1 und P2 berechnet sind, so können wir den zugehörigen Punkt P(x,y) auf der gestrichelten Kurve durch

```
x = xa + (xb-xa) * t
y = ya + (yb-ya) * t
```

erhalten. Für um 0 liegende t-Werte liegt ein Kurvenpunkt (x,y) nahe bei (xa,ya), und für t-Werte bei 1 liegt (x,y) nahe bei (xb,yb). Lediglich diese gemittelten Kurvenstücke (gestrichelte Kurve der Abb. 16) werden wir verwenden. Sind mehr Punkte vorhanden, so wird das obige Verfahren zunächst für P0,P1,P2,P3, dann weitergeschoben für P1,P2,P3,P4 usw. wiederholt. In dem folgenden Programm zur parabolischen Interpolation werden glatte Kurven durch die in DATA-Zeilen (1050,1060,1070,1110) abgelegten Punkte gezeichnet. Die Anzahl der vorgegebenen Punkte wird in n gespeichert.

Die Koordinaten der vorgegebenen Punkte werden in das x(),y()-Feld (Zeile 1200) gelesen. In den Zeilen 1220 bis 1250 wird ein Vorgängerpunkt P0 und ein nachfolgender (n+1)-ter Punkt gewählt. Die Zeilen 1350 bis 1410 markieren lediglich die gegebenen Punkte durch kleine Vierecke und können auch weggelassen werden. Die Zeilen 1450 bis 1640 folgen den oben angegebenen Betrachtungen. Für jeden t-Wert wird xa,ya (Zeilen 1500,1510) und xb,yb (Zeilen 1530,1540) berechnet. Mit xa,ya,xb,yb wird der Kurvenpunkt (x,y) berechnet (Zeilen 1560,1570). Durch die Zeilen 1590,1600 wird x,y auf die Bildschirmkoordinaten i2,j2 (0...639,0...399) transformiert. Die Maßstäbe mx,my wurden in der Zeile 1300 berechnet. In der Zeile 1620 wird eine Linie vom Punkt (x,y) mit dem „alten" t-Wert zum „neu" berechneten Kurvenpunkt gezeichnet. Ist die t-Schleife (Zeile 1450) abgearbeitet, so wird die Punktnummer k erhöht. Bei k=0 werden die Punkte P0,P1,P2 für die a-Parabel und P1,P2,P3 für die b-Parabel verwendet. Bei k=1 werden die Punkte P1,P2,P3 und P2,P3,P4 verwendet usw. Wenn wir die Zeilen 1560,1570 durch

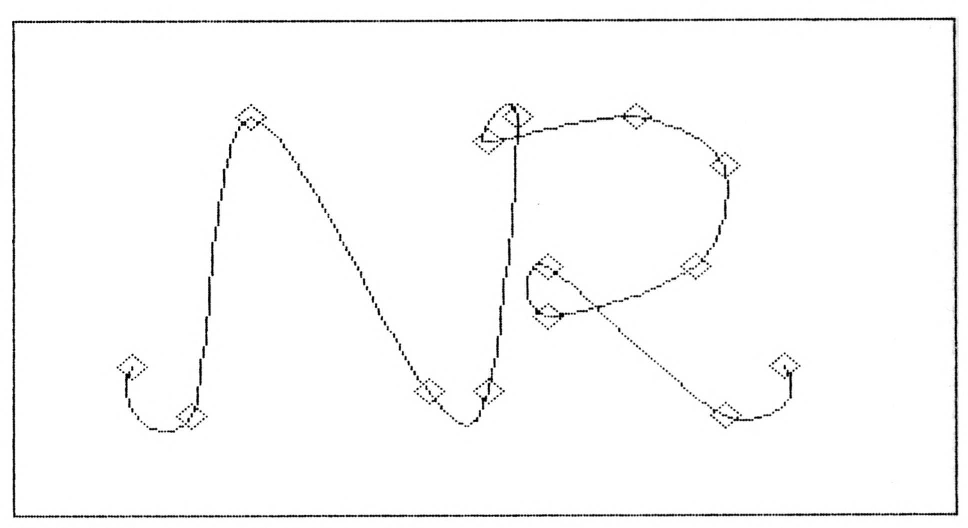
```
1560 x = xa + (xb - xa) * t*t * (3-2*t)
1570 y = ya + (yb - ya) * t*t * (3-2*t)
```

ersetzen, so ist neben der 1. Ableitung auch die 2. Ableitung des gesamten Kurvenzuges stetig.

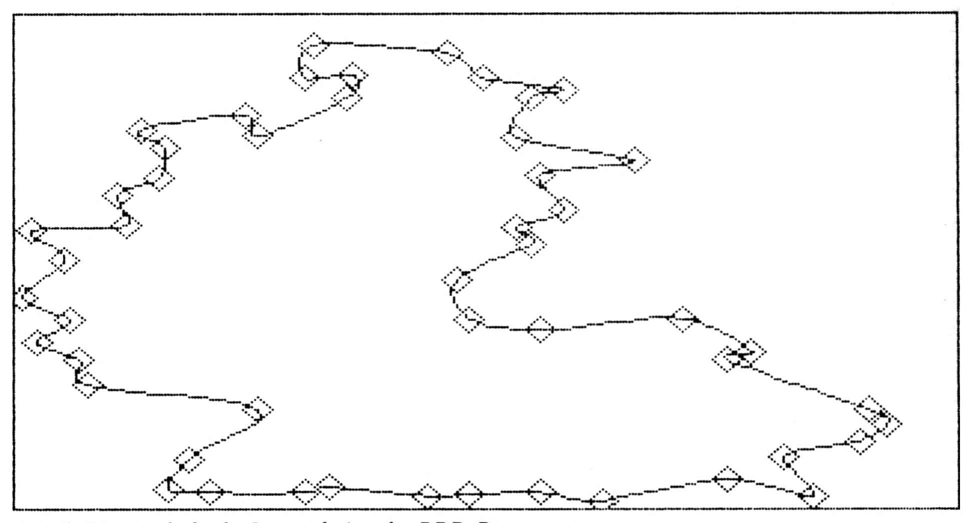
Mit dem gleichen Programm können auch geschlossene Kurven gezeichnet werden. In diesem Fall sind der erste und der n-te Punkt gleich. Außerdem ist der (n+1)-te Punkt gleich dem 1-ten Punkt (siehe Zeilen 1270,1280). Wenn wir die Zeilen 1270,1280,1281 aktivieren, so wird eine geschlossene Kurve durch die Punkte P1,P2,. . .,Pn gezeichnet. Entnehmen wir einem Atlas die Grenzpunkte der Bundesrepublik und legen diese in DATA-Zeilen ab, so ergibt sich die Grafik 57 (s. a. 13).

```
1020 REM--- parabolische Interpolation ---
1030 :
1040 REM   n, x1, y1,  x2, y2, .....  , xn, yn
1050 DATA 14, -6, -2,  -5, -3,  -4,  3, -1,-2.5
1060 DATA      0,-2.5,0.5,  3,   0,2.5,2.5,  3
1070 DATA      4,  2, 3.5, 0 ,   1, -1,  1,  0
1110 DATA      4, -3,   5, -2
1120 :
1130 DIM x(20), y(20)
1140 x8 =  -8: x9 =  8:  y8 = -5:  y9 =  5
1150 i8 = 0.5: i9=639.5: j8 = 0.5: j9 =399.5
1160 :
1170 :  READ n
1190 :  FOR k=1 TO n
1200 :   READ x(k), y(k)
1210 :  NEXT k
1215 :
1220 x(0)  = x(1)-x(2)+x(3)/3
1230 x(n+1)  = x(n)-x(n-1)+x(n-2)/3
1240 y(0)  = y(1)-y(2)+y(3)/3
1250 y(n+1)  = y(n)-y(n-1)+y(n-2)/3
1260 :
1270 REM--- x(0)=x(n): x(n+1)=x(1): x(n+2)=x(2)
1280 REM--- y(0)=y(n): y(n+1)=y(1)
1281 REM--- y(n+2)=y(2): n=n+1
1290 :
1300 mx = (i9-i8)/(x9-x8): my = (j9-j8)/(y9-y8)
1310 :
1320 MODE 1:BORDER 6:INK 0,1:INK 1,24
1340 :
1350 FOR k=1 TO n
1360 :  i2 = i8 + mx * (x(k) - x8)
1370 :  j2 = j8 + my * (y(k) - y8)
```

```
1380 :   REM--- IF k>1
                THEN MOVE i1,j1: DRAW i2,j2, 1
1390 :   MOVE i2-10,j2: DRAW i2,j2+10, 1
1395 :   DRAW i2+10,j2, 1: DRAW i2,j2-10, 1
1400 :   DRAW i2-10,j2, 1
1410 NEXT k
1420 :
1430 :
1440 FOR k=0 TO n-2
1450 :   FOR t=0 TO 1.01 STEP 0.1
1460 :
1470 :     h0 = (t-1)*t/2: h1 = 1-t*t
1480 :     h2 = (t+1)*t/2: h3 = (t-1)*(t-2)/2
1490 :     h4 = (2-t)*t
1495 :
1500 :     xa = h0*x(k) + h1*x(k+1) + h2*x(k+2)
1510 :     ya = h0*y(k) + h1*y(k+1) + h2*y(k+2)
1520 :
1530 :     xb = h3*x(k+1) + h4*x(k+2) + h0*x(k+3)
1540 :     yb = h3*y(k+1) + h4*y(k+2) + h0*y(k+3)
1550 :
1560 :     x = xa + (xb - xa) * t
1570 :     y = ya + (yb - ya) * t
1580 :
1590 :     i2 = i8 + mx * (x - x8)
1600 :     j2 = j8 + my * (y - y8)
1610 :
1620 :     IF t>0 THEN MOVE i1,j1: DRAW i2,j2, 1
1630 :     i1 = i2: j1 = j2
1640 :   NEXT t
1650 NEXT k
1670 a$=INKEY$: IF a$="" THEN 1670
```



Grafik 56: Stetige parabolische Interpolation



Grafik 57: Parabolische Interpolation der BRD-Begrenzung

14.3 Interpolation nach Newton

Bei der parabolischen Interpolation haben wir im vorhergehenden Kapitel das interpolierende Polynom nach LAGRANGE verwendet. Nun wollen wir die Formel von NEWTON einsetzen.

Sind z. B. in der t,y-Ebene die 3 Punkte P1(0,y1),P2(1,y2),P3(2,y3) gegeben, so geht durch diese Punkte das Polynom 2. Grades

```
yb = y1 + t * (y2 - y1 + (t - 1) * ((y1 + y3)/2 - y2)).
```

Für t=0 ist yb=y1. Für t=1 ist yb=y2 und für t=2 ist yb=y3. Durch die Punkte P0(−1,y0),P1(0,y1),P2(1,y2) geht das Polynom

```
ya = y0 + (t + 1) * (y1 - y0 + t * ((y0 + y2)/2 -
y1)).
```

Für t=−1 ist ya=y0. Für t=0 ist ya=y1 und für t=1 ist ya=y2. Verfahren wir analog mit xa und xb, so ergeben sich für t-Werte zwischen 0 und 1 das Polynom 3. Grades.

```
x = xa + t * (xb - xa)
y = ya + t * (yb - ya)
```

Im Kapitel 12 ist ein Programm zum Verbinden von Zufallspunkten durch glatte Kurven angegeben, das den obigen Betrachtungen entspricht.

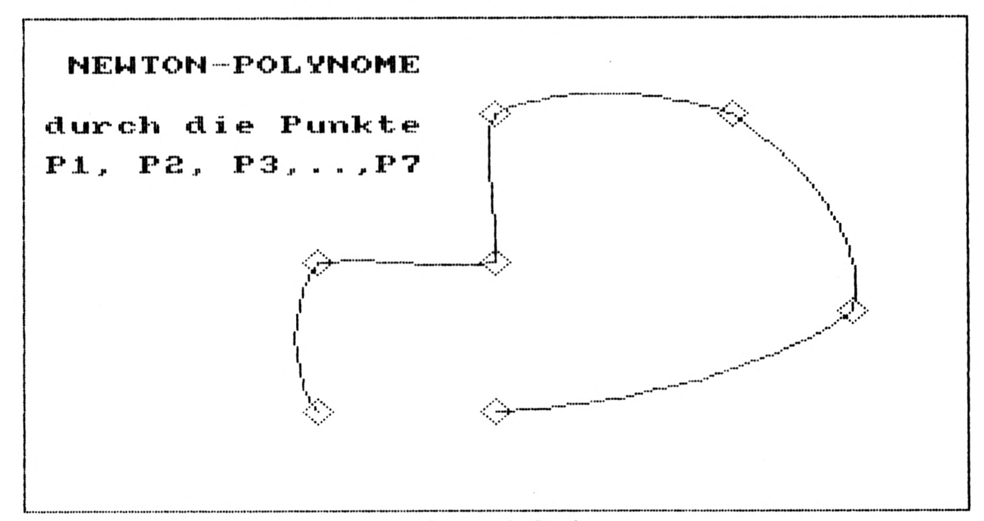
Wir wollen nun ein NEWTONSCHES Polynom 3. Grades angeben. In der t,y-Ebene seien die Punkte P0(−1,y0),P1(0,y1),P2(1,y2),P3(2,y3) gegeben. Mit den Abkürzungen

```
b1 = y1 - y0, b4 = (b2 - b1)/2
b2 = y2 - y1, b5 = (b3 - b2)/2
b3 = y3 - y2, b6 = (b5 - b4)/3
```

ergibt sich das Polynom 3. Grades.

```
y(t) = y0 + (t+1) * (b1 + t * (b4 + b6 * (t-1)).
```

Setzen wir t=−1 ein, so ergibt sich y=y0. Für t=0 ist y=y1. für t=1 ist y=y2, und für t=2 ist y=y3.



Grafik 58: Aneinandergesetzte Newton-Polynome 3. Grades

Analog verhält sich

$$x(t) = x0 + (t+1) * (a1 + t * (a4 + a6 * (t-1)))$$

mit a1=x1−x0,a2=x2−x1,a3=x3−x2,a4=(a2−a1)/2,a5=(a3−a2)/2,a6=(a5−a4)/3. Durchläuft t Werte von 0 bis 1, so ergeben die x(t),y(t) Formeln Punkte, die zwischen P1(x1,y1) und P2(x2,y2) liegen. Diese Formeln werden in dem folgenden Programm (Zeilen 1520 bis 1640) verwendet. Das Programm zeichnet durch aufeinanderfolgende Punkte interpolierende Kurven. Zum Zeichnen wird nur der mittlere Bereich von jeweils 4 aufeinanderfolgenden Punkten verwendet. An den Stoßstellen sind die Kurven nicht glatt, weil die Ableitungen rechts bzw. links von der Stoßstelle im allgemeinen verschieden sind.

```
1010 REM-       fortlaufende (stueckweise)      -
1020 REM-       parabolische Interpolation      -
1025 REM-               nach Newton             -
1030 REM- es treten (leider) spitze Ecken auf -
1035 :
1040 REM    n, x1,y1, x2,y2, ............, xn,yn
1050 DATA  7, -3,-3, -3, 0,  0, 0,  0, 3,  4, 3
1060 DATA      6,-1,  0,-3
1120 :
1130 DIM  x(20), y(20)
1140 x8 = -8:  x9 =  8:   y8 = -5:  y9 =  5
1150 i8 = 0.5: i9 =639.5: j8 = 0.5: j9 =399.5
1160 :
1170 :  READ n
1190 :  FOR k=1 TO n
1200 :   READ x(k),y(k)
1210 :  NEXT k
1215 :
1220 x(0)  = x(1)-x(2)+x(3)/3
1225 x(n+1)  = x(n)-x(n-1)+x(n-2)/3
1230 y(0)  = y(1)-y(2)+y(3)/3
1240 y(n+1)  = y(n)-y(n-1)+y(n-2)/3
1260 :
1270 REM--- x(0)=x(n): x(n+1)=x(1): x(n+2)=x(2)
1280 REM--- y(0)=y(n): y(n+1)=y(1): y(n+2)=y(2)
1281 REM--- n = n + 1
1290 :
1300 mx = (i9-i8)/(x9-x8): my = (j9-j8)/(y9-y8)
1310 :
1315 MODE 1:BORDER 6:INK 0,3:INK 1,26
1320 MOVE i8,j8: DRAW i9-i8,j8, 1
1325 DRAW i9-i8,j9-j8, 1: DRAW i8,j9-j8, 1
```

```
1327 DRAW i8,j8, 1
1330 LOCATE 3,3: PRINT "NEWTON-POLYNOME"
1335 LOCATE 2,6: PRINT "durch die Punkte"
1340 LOCATE 2,8: PRINT "P1, P2, P3,..,P7"
1345 :
1350 FOR k=1 TO n
1360 :   i2 = i8 + mx * (x(k) - x8)
1370 :   j2 = j8 + my * (y(k) - y8)
1380 :   REM--- IF k>1
                THEN MOVE i1,j1: DRAW i2,j2, 1
1385 :   MOVE i2-10,j2: DRAW i2,j2+10, 1
1390 :   DRAW i2+10,j2, 1: DRAW i2,j2-10, 1
1395 :   DRAW i2-10,j2, 1
1400 NEXT k
1405 :
1410 a2=x(1)-x(0): a3=x(2)-x(1): a5=(a3-a2)/2
1420 b2=y(1)-y(0): b3=y(2)-y(1): b5=(b3-b2)/2
1430 :
1440 FOR k=0 TO n-2
1460 : a1 = a2: a2 = a3: a3 = x(k+3)-x(k+2)
1470 : b1 = b2: b2 = b3: b3 = y(k+3)-y(k+2)
1480 : a4 = a5: a5 = (a3-a2)/2: a6 = (a5-a4)/3
1490 : b4 = b5: b5 = (b3-b2)/2: b6 = (b5-b4)/3
1500 : a0 = x(k): b0 = y(k)
1510 :
1520 : FOR t=0 TO 1.01 STEP 0.1
1540 :     x = (((t-1)*a6+a4)*t+a1)*(t+1) + a0
1550 :     y = (((t-1)*b6+b4)*t+b1)*(t+1) + b0
1580 :
1590 :     i2 = i8 + mx * (x-x8)
1600 :     j2 = j8 + my * (y-y8)
1610 :
1620 :     IF t>0 THEN MOVE i1,j1: DRAW i2,j2, 1
1630 :     i1 = i2: j1 = j2
1640 : NEXT t
1650 NEXT k
1660 :
1670 a$=INKEY$: IF a$="" THEN 1670
```

Die Berechnung der Koeffizienten a1,a4,a6,b1,b4,b6 erfolgt außerhalb der t-Schleife in den Zeilen 1460 bis 1500. Dadurch ist der Zeitbedarf des Programmes gering. Als Nachteil ist zu nennen, daß beim „Weiterschieben" der Formeln zu den nächsten 4 Stützpunkten (Zeile 1440 ff.) zwar ein stetiger, aber nicht glatter Anschluß erfolgt; d. h., es können „Knicke" in

den Stützpunkten auftreten (Grafik 58). Durch die Zeilen 1350 bis 1400 werden lediglich die vorgegebenen Stützpunkte durch Quadrate markiert. In den Zeilen 1220 bis 1240 wird ein Vorgängerpunkt P0(x0,y0) und ein Nachfolgepunkt Pn+1 (zusätzlich zu den n-Stützpunkten P1(x1,y1),. . .,Pn(xn,yn)) definiert. Soll der gesamte Kurvenzug geschlossen werden, so ist der n-te Punkt mit dem ersten Punkt zu verbinden. In diesem Fall sind die Zeilen 1270,1280,1281 zu aktivieren.

14.4 Rationale Interpolation

Wir wollen hier einen speziellen Algorithmus zur rationalen Interpolation einführen, der sich durch wenige Rechenoperationen auszeichnet. Die allgemeine Theorie zu der Interpolation mit rationalen Funktionen ist z. B. in dem Buch „Stoer: Einführung in die Numerische Mathematik, Springer-Verlag" enthalten.

Wir betrachten zunächst die Abb. 17. In der t,y-Ebene seien die Punkte P1(0,y1) und P2(1,y2) sowie die Steigungen $s1=\tan(\beta 1)$,$s2=\tan(\beta 2)$ der Tangenten gegeben. Diese Tangenten berühren die gesuchte (strichpunktierte) Linie in P1 bzw. P2. Die Sekante durch P1,P2 kann mit der Parameterdarstellung der Geraden durch

$$y = y1 + t * (y2 - y1)$$

ausgedrückt werden. Für jeden t-Wert zwischen 0 und 1 können Sekantenpunkte berechnet werden. Insbesondere sei Q(t,y) berechnet (Zeile 1535). Die Abweichungen b1,b2 der Tangenten von der Sekanten sind ein Maß für die Abweichung der gesuchten Kurve (strichpunktiert) von der Sekanten. Mit Hilfe des linken schraffierten Dreieckes (P1,Q1,Q3) berechnen wir die Abweichung b1. Die senkrechte Strecke von Q1 bis Q3 ist $t*\tan(\beta 1) = y-y1+b1$, d. h.

$$b1 = y1 - y + t * s1.$$

In ähnlicher Weise ergibt sich aus dem rechten schraffierten Dreieck (P2,Q2,Q4)

$$b2 = y2 - y - (1 - t) * s2.$$

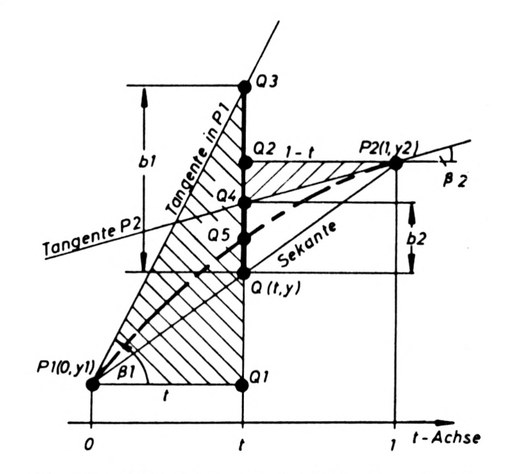


Abb. 17: Abweichungen b1,b2 *für die rationale Interpolation*

In der Zeile 1545 werden b1,b2 berechnet. Wenn b1 oder b2 gleich Null ist, so fallen die Tangente und die Sekante zusammen. In diesem Falle, wenn $b=b1*b2=0$ ist, entspricht die gesuchte Kurve der Sekanten $y=yt+t*(y2-y1)$. Ist $b1*b2>0$, so liegt eine geometrische Anordnung ähnlich der Abb. 17 vor, und wir verbessern den y-Wert der Sekanten durch Addition von $b1*b2/(b1+b2)$. Diese Verbesserung wird in der Zeile 1565 durchgeführt. Ist $b1*b2<0$, so liegen b1 und b2 auf verschiedenen Seiten der Sekante. Die gesuchte Kurve

wendet in dem betrachteten t-Bereich (Zeile 1570). In diesem Falle wird der y-Wert der Sekannte durch Addition von b1*b2/(b1−b2)*(2t−1) verbessert.

Anstelle eines t,y-Koordinatensystems können wir in gleicher Weise die obigen Betrachtungen in einem t,x-Koordinatensystem durchführen. Die Steigungen sind mit r1,r2 und die Abweichungen von der gesuchten Kurve mit a1,a2 bezeichnet (Zeilen 1530,1540,1555,1560). Bei der hier vorgeschlagenen Methode ist die Wahl der Steigungen in den vorgegebenen Punkten frei. Betrachten wir z. B. in der t,y-Ebene 3 aufeinanderfolgende Punkte P0(−1,y0),P1(0,y1),P2(1,y2), so können wir als Steigung s1 im Punkt P1 den Wert

```
s1 = (y2 - y0)/2
```

wählen. Diese Wahl entspricht der Steigung der Parabel durch P0,P1,P2 im Punkt P1. In den Zeilen 1430,1510 werden diese Steigungen berechnet. Gute Ergebnisse ergeben sich, wenn ein Kreis verwendet wird, der durch 3 aufeinanderfolgende Punkte P0,P1,P2 geht. Die Steigung s1 im Punkt P1 ist dann mit

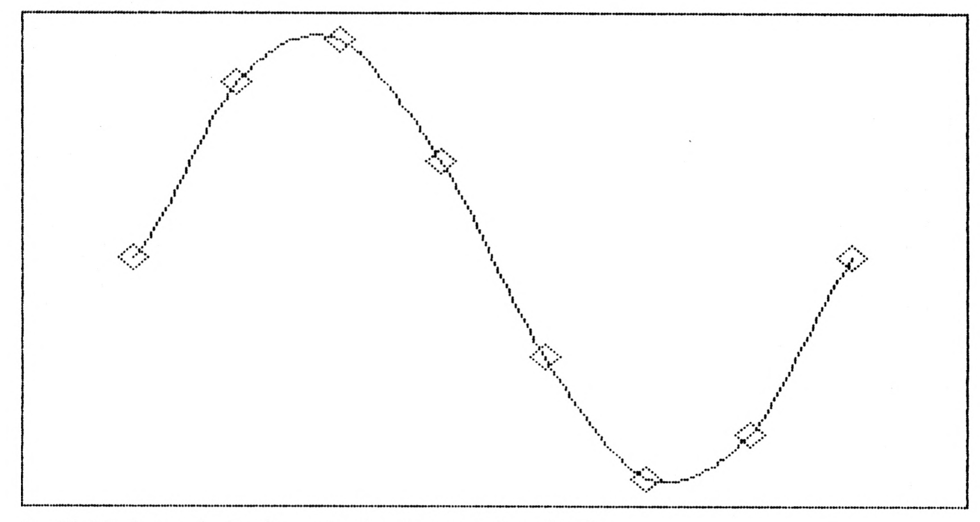
```
h1 = 1 + (y1 - y0) * (y1 - y0)
h2 = 1 + (y2 - y1) * (y2 - y1)
```

durch

```
s1 = ((y1 - y0) * h2 + (y2 - y1) * h1)/(h2 + h1)
```

gegeben.

Die Grafik 59 ergibt sich als Interpolationskurve durch 8 vorgegebene Punkte der Sinus-Linie (Zeilen 1170 bis 1210). In den Zeilen 1220,1230 werden die Koordinaten eines vorhergehenden und nachfolgenden Punktes gewählt. Als Steigung des mittleren von 3 aufeinanderfolgenden Punkten wird die Parabel-Steigung gewählt.



Grafik 59: Interpolationskurve durch acht vorgegebene Punkte

```
1010 REM fortlaufende rationale Interpolation
1035 :
1130 DIM x(20), y(20)
1140 x8 = -1: x9 = 7.3: y8 = -1.1: y9 = 1.1
1150 i8 = 0.5: i9 =639.5: j8 = 0.5: j9 =399.5
1160 :
1170 :  n = 8
1190 :  FOR k=1 TO n
1200 :  x(k) = (k-1)*2*PI/(n-1)
1205 :  y(k) = SIN(x(k))
1210 :  NEXT k
1215 :
1220 x(0)=x(1)-x(2)+x(3)/3:
     x(n+1)=x(n)-x(n-1)+x(n-2)/3
1230 y(0)=y(1)-y(2)+y(3)/3:
     y(n+1)=y(n)-y(n-1)+y(n-2)/3
1260 :
1300 mx = (i9-i8)/(x9-x8): my = (j9-j8)/(y9-y8)
1310 :
1320 MODE 1:BORDER 2:INK 0,1:INK 1,24
1330 MOVE i8,j8: DRAW i9-i8,j8, 1
1335 DRAW i9-i8,j9-j8, 1: DRAW i8,j9-j8, 1
1340 DRAW i8,j8, 1
1345 :
1350 FOR k=1 TO n
1360 :  i2 = i8 + mx * (x(k) - x8)
1370 :  j2 = j8 + my * (y(k) - y8)
1380 :  REM--- IF k>1
                THEN MOVE i1,j1: DRAW i2,j2, 1
1390 :  MOVE i2-10,j2: DRAW i2,j2+10, 1
1395 :  DRAW i2+10,j2, 1: DRAW i2,j2-10, 1
1400 :  DRAW i2-10,j2, 1
1410 NEXT k
1420 :
1430 r2=(x(2)-x(0))/2: s2=(y(2)-y(0))/2
1435 :
1440 FOR k=1 TO n-1
1505 :  r1=r2: r2=(x(k+2)-x(k))/2
1510 :  s1=s2: s2=(y(k+2)-y(k))/2
1515 :
1520 :  FOR t=0 TO 1.01 STEP 0.05
1530 :     x = x(k) + t*(x(k+1)-x(k))
```

```
1535 :       y = y(k)  + t*(y(k+1)-y(k))
1540 :       a1=x(k)  - x + t*r1:
             a2=x(k+1)  - x -  (1-t)*r2
1545 :       b1=y(k)  - y + t*s1:
             b2=y(k+1)  - y -  (1-t)*s2
1550 :       a = a1*a2: b = b1*b2
1555 :       IF a>0 THEN x = x+a/(a1+a2)
1560 :       IF a<0 THEN x = x+a/(a1-a2)*(t+t-1)
1565 :       IF b>0 THEN y = y+b/(b1+b2)
1570 :       IF b<0 THEN y = y+b/(b1-b2)*(t+t-1)
1580 :
1590 :       i2 = i8 + mx *  (x-x8)
1600 :       j2 = j8 + my *  (y-y8)
1610 :
1620 :       IF t>0 THEN MOVE i1,j1: DRAW i2,j2, 1
1630 :       i1 = i2: j1 = j2
1640 :   NEXT t
1650 NEXT k
1660 :
1670 a$=INKEY$: IF a$="" THEN 1670
```

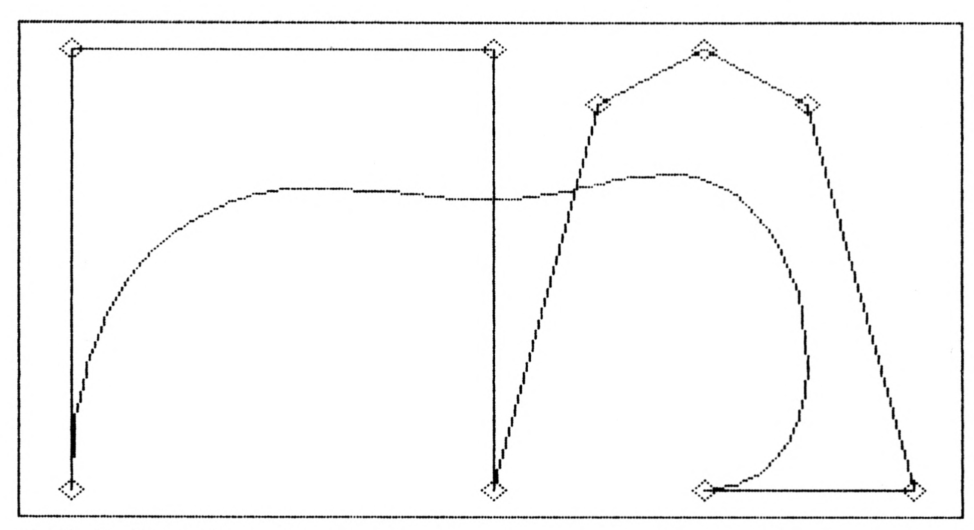
14.5 Bezier-Kurven

BEZIER-Kurven verhalten sich ähnlich einem biegesteifen Stahlband, dessen Enden fest eingespannt sind, wobei zusätzlich eine Dehnbarkeit des Stahlbandes in Längsrichtung angenommen wird.

Zieht man beispielsweise das Stahlband in die Mitte nach unten, so tritt eine Verformung auf. Aufgrund der Elastizität treten keine Knicke, sondern „gerundete“ Ausbeulungen auf. An dem Stahlband- Modell könnten noch weitere Kräfte in unterschiedlicher Richtung angreifen. Auf diese Weise würden glatte Kurven ohne Ecken und Kanten entstehen. Diese Kurven entsprechen BEZIER-Kurven. Die vorgegebenen Stützstellen stellen Zug-Angriffspunkte dar, unter denen sich das Stahlband wie eine „Gummihaut“ wölbt. Bis auf die Endpunkte liegen gewöhnlich alle Stützstellen außerhalb der BEZIER-Kurve.

BEZIER entwickelte für die französische Autofirma Renault diese Methode zur Gestaltung von gekrümmten Automobilteilen mit Hilfe der Computer-Grafik.

Sind x(k),y(k) mit k=0,1,2,. . .,n die vorgegebenen Stützstellen, so ergeben sich die Kurvenpunkte x(t),y(t) gemäß der Binominal-Entwicklung:



Grafik 60: Bezier-Kurve

$$x(t) = \sum_{k=0}^{n} \binom{n}{k} * t^k * (1-t)^{n-k} * x(k)$$

$$y(t) = \sum_{k=0}^{n} \binom{n}{k} * t^k * (1-t)^{n-k} * y(k)$$

Der Binominalkoeffizient

$$\binom{n}{k} = \frac{n*(n-1)*(n-2)*\ldots\ldots(n-k+1)}{1*2*3*\ldots k}$$

ist im Programm mit c(k) bezeichnet und wird in der Schleife 440—460 berechnet. Für jeden Parameter t, der zwischen 0 und 1 liegt, muß der Kurvenpunkt x(t),y(t) berechnet werden. Hierzu ist jeweils über alle Punkte (Zeile 540 bis 580) gemäß der obigen Formeln zu summieren. Diese Summation wird in den x,y-Speichern durchgeführt (Zeilen 560,570). Die Koordinaten x,y werden in den Zeilen 600,610 in die entsprechenden Bildschirmkoordinaten umgerechnet. In der Zeile 620 wird eine Linie vom vorhergehenden Kurvenpunkt (i1,j1) zum aktuellen Kurvenpunkt (i2,j2) gezeichnet. Der erste Kurvenpunkt (für t=0) wird außerhalb der t-Schleife (Zeilen 520—640) in den Zeilen 480,490 berechnet. Die Schrittweite d (Zeile 510) bestimmt die Anzahl der zu berechnenden Kurvenpunkte gemäß der Zeile 520. Die Zeilen 330 bis 410 dienen lediglich zum Zeichnen und Verbinden der n vorgegebenen Kontrollpunkte x(k),y(k) durch einen Polygonzug. Die BEZIER-Kurven bilden Schleifen, wenn sich die vorgegebenen Kontrollpunkte x(k),y(k) überkreuzen. Wird ein Punkt mehrfach eingegeben, so nähert sich die BEZIER-Kurve diesem Punkt stärker an.

```
150 REM--- Bezier-Kurven ---
160 :
170 REM  n, x0,y0, x1,y1, x2,y2, ... , xn,yn
180 DATA 8, -4,-4, -4, 4,  0, 4, 0,-4,  1, 3
190 DATA       2, 4,  3, 3,  4,-4, 2,-4
200 :
210 x8 =-4.5: x9 =  4.5: y8 =-4.5: y9 =  4.5
220 i8 = 0.5: i9 =639.5: j8 = 0.5: j9 =399.5
230 :
240 READ n: DIM x(n), y(n), c(n)
250 FOR k=0 TO n: READ x(k),y(k): NEXT k
260 :
270 mx = (i9-i8)/(x9-x8): my = (j9-j8)/(y9-y8)
280 MODE 1:BORDER 2:INK 0,1:INK 1,24
290 MOVE i8,j8: DRAW i9-i8,j8, 1
300 DRAW i9-i8,j9-j8, 1: DRAW i8,j9-j8, 1
310 DRAW i8,j8, 1
320 :
330 FOR k=0 TO n
340 :   i2 = i8 + mx * (x(k) - x8)
350 :   j2 = j8 + my * (y(k) - y8)
360 :   IF k>0 THEN MOVE i1,j1: DRAW i2,j2, 1
370 :   MOVE i2-8,j2: DRAW i2,j2+8, 1
380 :   DRAW i2+8,j2, 1: DRAW i2,j2-8, 1
390 :   DRAW i2-8,j2, 1
400 :   i1 = i2: j1 = j2
410 NEXT k
420 :
430 c(0) = 1
440 FOR k=1 TO n
450  c(k) = c(k-1)*(n-k+1)/k
460 NEXT k
470 :
480 i1 = i8 + mx * (x(0) - x8)
490 j1 = j8 + my * (y(0) - y8)
500 :
510 d = 0.02
520 FOR t=d TO 1+d/2 STEP d: x = 0: y = 0
530 :   IF t>=1 THEN x = x(n): y = y(n):GOTO 600
540 :   FOR k=0 TO n
550 :       c = c(k) * t^k * (1-t)^(n-k)
560 :       x = x + c*x(k)
```

```
570 :     y = y + c*y(k)
580 :   NEXT k
590 :
600 :   i2 = i8 + mx * (x-x8)
610 :   j2 = j8 + my * (y-y8)
620 :   MOVE i1, j1: DRAW i2, j2, 1
630 :   i1 = i2: j1 = j2
640 NEXT t
650 :
660 a$=INKEY$: IF a$="" THEN 660
```

15. 3D-Darstellungen

Dreidimensionale grafische Darstellungen vermitteln dem Beobachter im allgemeinen einen natürlichen Seheindruck. Zur besseren Übersicht werden verdeckte Linien und Kanten durch einen geeigneten HIDDEN-LINE-Algorithmus unterdrückt. Wir besprechen zunächst die Dimetrie, Isometrie, Kavalierperspektive und die Militärperspektive und leiten unter anderem die benötigten Formeln ab.

15.1 Parallelprojektionen

Wir wollen nun 3-dimensionale Objekte auf dem Bildschirm darstellen und die zugehörigen Abbildungsformeln herleiten. Dazu betrachten wir die Abb. 18. Ein Raumpunkt P ist durch die 3 Koordinaten u,v,w festgelegt. Die Zeichenebene (Bildschirmebene) sei die x,y-Ebene. Der Raumpunkt P(u,v,w) hat in der Zeichenebene die Koordinaten x,y. Unser Ziel ist es, bei vorgegebenen u,v,w-Werten die x,y-Werte zu ermitteln. Betrachten wir in der Abb. 18 die dicken, horizontalen Linien, so ergibt sich x+x2=x1. Die dicken vertikalen Linien liefern y+y1+y2=w.

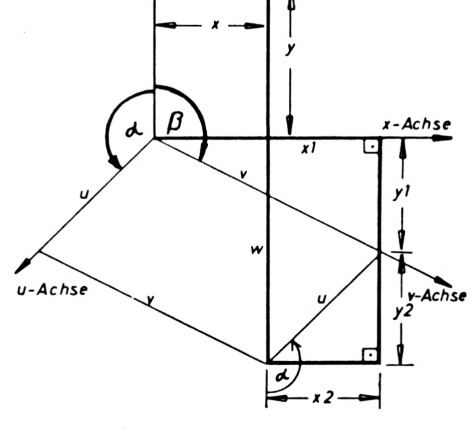


Abb. 18: 3D-Abbildung des Raumpunktes P(u,v,w) in der x,y-Ebene

Somit ist

$$x = x1 - x2$$
$$y = w - y1 - y2.$$

Wenn wir x1,x2,y1,y2 durch u,v ausdrücken, so haben wir die gesuchten Formeln für das 3D-Zeichnen. Das x1,y1,v-Dreieck liefert

$$x1 = v * \cos(\beta-90°) = +v * \sin(\beta)$$
$$y1 = v * \sin(\beta-90°) = -v * \cos(\beta),$$

und das x2,y2,u-Dreieck liefert

$$x2 = u * \cos(\alpha-90°) = +u * \sin(\alpha)$$
$$y2 = u * \sin(\alpha-90°) = -u * \cos(\alpha).$$

Setzen wir x1,x2,y1,y2 in die Formeln x=x1−x2, y=w−y1−y2 ein, so erhalten wir die Endformeln

$$x = -u*\sin(\alpha) + v*\sin(\beta)$$
$$y = +u*\cos(\alpha) + v*\cos(\beta) + w.$$

Diese leicht zu programmierenden Endformeln bilden die Grundlage für alle 3D-Parallelprojektionen, wie z. B. die Dimetrie, Isometrie, Kavalierperspektive, Militärperspektive.

Werden für u,v,w die Koordinaten von Körperpunkten verwendet, so liefern die Endformeln die zugehörigen Koordinaten x,y in der Zeichenebene. Mit diesen beiden Formeln können alle Eckpunkte eines Körpers in Zeichenkoordinaten umgerechnet werden. Durch Auftragen dieser x,y-Werte erhalten wir in der Zeichenebene ein 3-dimensionales Bild („Drahtmodell“) des Körpers.

15.2 Dimetrie, Isometrie, Kavalier- und Militärperspektive

Wir wollen diese allgemeinen Gleichungen durch die Wahl von α,β spezialisieren. Für die Dimetrie gilt $\alpha=131.42°$ und $\beta=97.18°$. Weiterhin ist für die Dimetrie festgelegt, daß die u-Werte (s. Abb. 18) verkürzt werden, d. h. es gilt u:v:w = 0.5:1:1.

Setzen wir dies in $x=-u*\sin(\alpha)+v*\sin(\beta)$ und $y=+u*\cos(\alpha)+v*\cos(\beta)+w$ ein, so ergeben sich die **Dimetrie**-Formeln

```
x = -0.37494 * u + 0.99216 * v
y = -0.33079 * u - 0.12499 * v + w.
```

Zur Berechnung der Zeichenkoordinaten x,y aus den Eckpunkten (u,v,w) eines Körpers kann z. B. das folgende Programm verwendet werden:

```
10 INPUT "u,v,w ="; u, v, w
20 x = - 0.37494 * u + 0.99216 * v
30 y = - 0.33079 * u - 0.12499 * v + w
40 PRINT x, y
50 GOTO 10
```

Geben wir z. B. die Eckpunkte (u,v,w) eines Würfels (s. folgende Tabelle) mit der Kantenlänge 100 ein, so erhalten wir die x,y-Werte:

u	v	w	x	y
0	0	0	0.0	0.0
100	0	0	−37.4	−33.0
100	100	0	61.7	−45.5
0	100	0	99.2	−12.4
0	0	100	0.0	100.0
100	0	100	−37.4	66.9
100	100	100	61.7	54.4
0	100	100	99.2	87.5

Tragen wir die x,y-Werte in ein x,y-Koordinatensystem ein und verbinden die Punkte, so ergibt sich die Abb. 19. Bei der dimetrischen Darstellung von Körpern erscheinen senkrechte Kanten auch im Bild senkrecht. Die Darstellung des Würfels in der Abb. 19 zeigt, daß wir „von vorn" gegen die v,w-Ebene des Würfels sehen.

Soll statt dessen von vorn die u,v-Ebene des Würfels sichtbar sein, so ist das u,v,w-System um 90° um die w-Achse zu drehen. Mit $\alpha=262.82°$, $\beta=48.58°$ und u:v:w = 1:0.5:1 ergeben sich aus $x=-u*\sin(\alpha) + v*\sin(\beta)$, $y=+u*\cos(\alpha) + v*\cos(\beta) + w$ die **Dimetrie**-Formeln:

```
x = +0.99216 * u + 0.37494 * v
y = -0.12499 * u + 0.33079 * v + w.
```

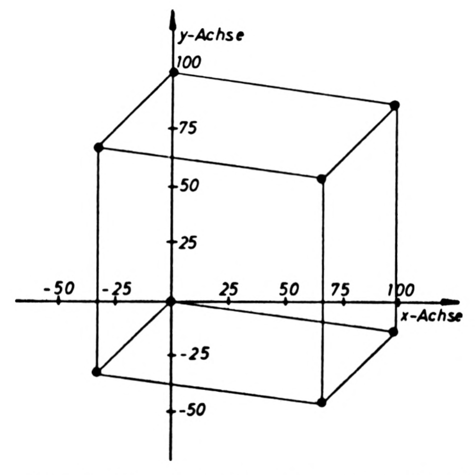


Abb. 19: Dimetrische Darstellung eines Würfels

Für andere Parallelprojektionen sind die Winkel α,β in den allgemeinen Formeln $x=-u*\sin(\alpha) + v*\sin(\beta)$, $y=+u*\cos(\alpha) + v*\cos(\beta) + w$ entsprechend zu spezifizieren. Setzen wir a=120°, $\beta=120°$, so ergeben sich die

Isometrie-Formeln

```
x = (-u+v) * sqr(3)/2
y = (-u-v)/2 + w.
```

Für die **Kavalierperspektive** gilt $\alpha = 135°$, $\beta = 90°$ und mit u:v:w = 0.5:1:1 ergibt sich

```
x = -0.35355 * u + v
y = -0.35355 * u + w.
```

Bei der Kavalierperspektive werden alle zum Aufriß parallelen Flächen in wahrer Größe abgebildet.

Für die **Militärperspektive** gilt $\alpha = 135°$, $\beta = 135°$ und mit u:v:w = 1:1:1 ergibt sich

```
x = (-u+v) * sqr(2)/2
y = (-u-v) * sqr(2)/2 + w.
```

Bei der Militärperspektive werden alle zum Grundriß parallelen Flächen in wahrer Größe abgebildet. Die lotrechten Strecken sind untereinander parallel und erscheinen in wahrer Größe.

15.3 3D-Darstellungen von Funktionen

Wir wollen Funktionen w=f(u,v), die über einem rechteckigen Gebiet u8<=u<=u9,v8<=v<=v9 erklärt sind, grafisch darstellen. Zur Erläuterung wählen wir die Funktion $w=\cos(u)*e^{-v*v}$, die in der BASIC-Schreibweise

```
w = COS(u) * EXP(-v*v)
```

entspricht.

Der jeweilige w-Wert ist durch die Variablen u,v festgelegt. Wenn wir „per Hand" solche **Funktionen, die von 2 Variablen** abhängen, aufzeichnen wollen, so ist zunächst eine Wertetabelle zu berechnen. Hierzu verwenden wir 2 geschachtelte Schleifenanweisungen (Zeilen 510,520).

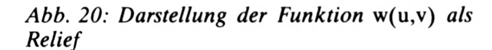
```
130 REM--- Wertetabelle fuer w = f(u,v) ---
210 u8 = -2:  u9 =   2:   v8 = -2:  v9 = 2
220 i8 =0.5:  i9 =639.5:  j8 =0.5:  j9 =399.5
260 x8 = -3:  x9 =   3:   y8 = -1:  y9 = 1.2
265 :
450 mx = (i9-i8)/(x9-x8)
460 my = (j9-j8)/(y9-y8)
480 :
510 FOR v=v8 TO v9 STEP (v9-v8)/4
520 :   FOR u=u8 TO u9 STEP (u9-u8)/4
530 :
540 :     w = EXP(-v*v)  * COS(u)
545 :
550 :     x = u + v/SQR(8)
555 :     y = w + v/SQR(8)
560 :     i2% = i8 + mx * (x-x8)
570 :     j2% = j8 + my * (y-y8)
575 :
610 :     PRINT u;  v;  w;  x;  y,  i2%;  j2%
615 :
700 :   NEXT u
710 NEXT v
```

Der Wert der Funktion wird in Zeile 540 berechnet. Da die u-Schleife als innere Schleife besonders oft durchlaufen wird, ist darauf zu achten, daß unnötige Berechnungen aus der inneren Schleife herausgenommen werden.

Z. B. könnte in der Zeile

```
515 h = EXP(-v*v)
```

berechnet und in der Zeile 540 durch



```
540 w = h * COS(u)
```

ersetzt werden. Obwohl durch das Entfernen von Termen aus der inneren Schleife die Ausführungszeit verkürzt wird, verzichten wir an dieser Stelle zugunsten der besseren Übersicht darauf.

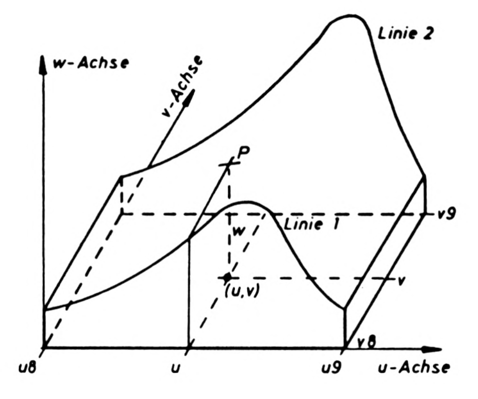


Abb. 20: Darstellung der Funktion w(u,v) *als Relief*

Für die Funktion w=w(u,v) bilden die vorgegebenen u-Grenzen u8,u9 und die v-Grenzen v8,v9 (s. Zeile 210) ein rechteckiges Grundgebiet. In der Abb. 20 ist die Funktion w(u,v) symbolisch als Relief dargestellt. Zu einem Punkt der u,v-Ebene kann die Stützhöhe w=w(u,v) berechnet werden. Diese Stützhöhen bilden eine (gekrümmte) Relief-Fläche im Raum. Jeder Punkt p dieser Fläche wird durch die räumlichen Koordinaten u,v,w beschrieben. Die Umrechnung dieser Raumpunkte (u,v,w) in die Zeichenebene (x,y-Ebene) ist im Kapitel 15.1 erklärt. Für die Kavalierperspektive gelten die Formeln:

```
x = u + v/sqr(8)
y = w + v/sqr(8)
```

Hierbei ist (u,v,w) ein Raumpunkt und (x,y) der zugehörige Punkt in der Zeichenebene. Durch die Programmzeilen 510,520,540 sind u,v,w gegeben. Die Umrechnung in die Bildschirmkoordinaten wird durch die Zeilen 560,570 veranlaßt. In der Zeile 610 wird u,v,w und x,y ausgegeben. Abgesehen von der Form der Ausgabe liefert das Programm für die Stützhöhen w der Funktion w = COS(u) * EXP(−v*v) die Werte:

v	u	−2.00	−1.00	0.00	1.00	2.00
−2.0		−0.01	0.01	0.02	0.01	−0.01
−1.0		−0.15	0.20	0.37	0.20	−0.15
0.0		−0.42	0.54	1.00	0.54	−0.42
+1.0		−0.15	0.20	0.37	0.20	−0.15
+2.0		−0.01	0.01	0.02	0.01	−0.01

Es kann z. B. für v=−1,u=0 aus der obigen Tabelle der Wert w=0.37 entnommen werden. Als zugehöriger Punkt (x,y) in der Zeichenebene liefert die folgende Tabelle x=−0.35,y=0.01 für die Koordinaten x,y in der Zeichenebene (Kavalierperspektive x=u+v/sqr(8), y=w+v/sqr(8)).

v	u −2.00	−1.00	0.00	1.00	2.00
−2.0	−2.71/−0.71	−1.71/−0.70	−0.71/−0.69	+0.29/−0.70	+1.29/−0.71
−1.0	−2.35/−0.51	−1.35/−0.15	−0.35/+0.01	+0.65/−0.15	+1.65/−0.51
0.0	−2.00/−0.42	−1.00/+0.54	0.00/+1.00	+1.00/+0.54	+2.00/−0.42
+1.0	−1.65/+0.20	−0.65/+0.55	+0.35/+0.72	+1.35/+0.55	+2.35/+0.20
+2.0	−1.29/+0.70	−0.29/+0.72	+0.71/+0.73	+1.71/+0.72	+2.71/+0.70

Wenn wir die Punkte (x,y) der letzten Tabelle in ein x,y-System eintragen und die Punkte durch Geraden verbinden, so erhalten wir Abb. 21.

Indem wir eine kleinere u- und v-Schrittweite (Zeile 510,520) wählen, ergeben sich mehr Bildpunkte und damit eine genauere Darstellung von w = COS(u) * EXP(−v*v).

Zur besseren Verständlichkeit wurde die tabellarische Darstellung von w = w(u,v) und das Zeichnen von Hand ausgeführt. Die Darstellung von Funktionen mit zwei Variablen in einer Tabelle ist in der Technik weit verbreitet.

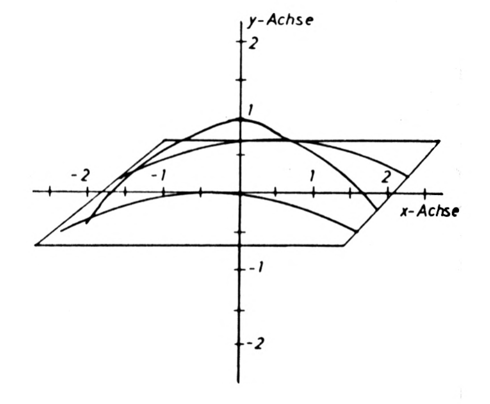


Abb. 21: Darstellung der Zeichenebene „per Hand“

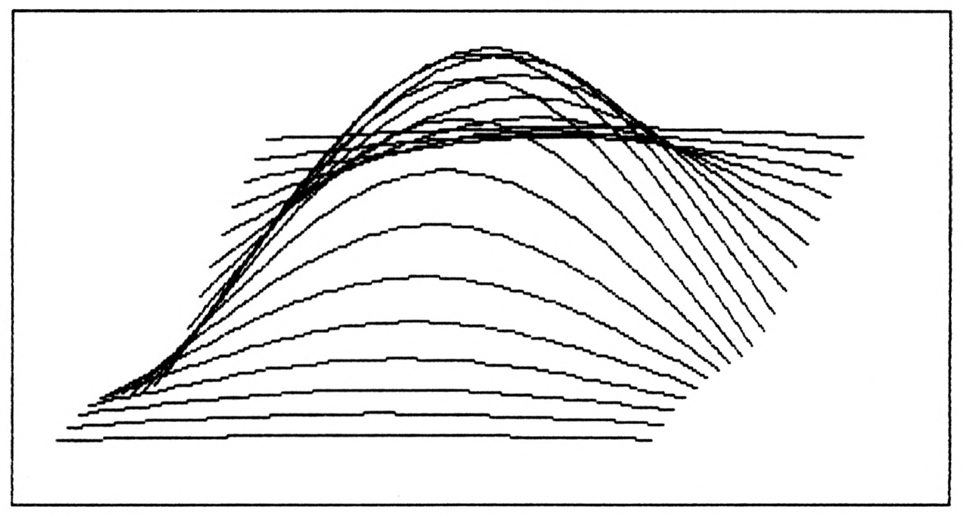
Damit das Wertetabellenprogramm die Funktion w = COS(u) * EXP(−v*v) automatisch zeichnet, ist es zu ergänzen. Die Programmzeile 220 legt den linken (i8), rechten (i9), unteren (j8) und oberen (j9) Rand des Bildschirmes fest. Die Umrechnung von x,y in die physikalischen Bildpunkte (i2%,j2%) geschieht durch die Zeilen 560,570. Da bei der Besetzung der Ganzzahlvariablen (Integer) i2%,j2% die Nachkommastellen abgeschnitten werden, wurde 0.5 (s. Zeile 220) addiert.

Die Werte von i2% sind ganzzahlig und müssen zwischen 0 und 639 liegen. Entsprechend liegt j2% zwischen 0 und 399. Die x,y-Maßstäbe mx,my für die Umrechnung von x,y in i2%,j2% werden in den Zeilen 450,460 ermittelt und in den Zeilen 560,570 verwendet. Damit für zwei aufeinanderfolgend berechnete Punkte die Verbindungslinie gezeichnet wird, sind die Zeilen 265,610,615 durch:

```
265 MODE 2:BORDER 3:INK 0,1:INK 1,26
610 IF u<>u8 THEN MOVE i1%,j1%: DRAW i2%,j2%,1
615 i1% = i2%: j1% = j2%
```

zu ersetzen. Wird die Schrittweite (Zeilen 510,520) verkleinert, so liefert der Programmlauf die Grafik 61. Hierbei ergibt sich für jeden v-Wert eine Linie, die der Funktion w=w(u,v) mit v als Parameter entspricht (siehe Reliefdarstellung Abb. 20). Beim Zeichnen wird keine

Rücksicht darauf genommen, ob Linien unsichtbar sind. Zur besseren Übersicht geben wir das geänderte Programm vollständig wieder:



Grafik 61: 3D-Darstellung von w = EXP(−v*v) * COS(u) *einschließlich der nichtsichtbaren Linien*

```
130 REM--- 3-D-Darstellung von Funk-   ---
140 REM--- tionen  w = w(u,v) mit      ---
150 REM--- allen unsichtbaren Linien   ---
210 u8 = -2:  u9 =  2:   v8 = -2:  v9 = 2
220 i8 =0.5:  i9 =639.5: j8 =0.5:  j9 =399.5
260 x8 = -3:  x9 =  3:   y8 = -1:  y9 = 1.2
263 :
265 MODE 2:BORDER 3:INK 0,1:INK 1,26
270 MOVE i8,j8: DRAW i9-i8,j8, 1
275 DRAW i9-i8,j9-j8, 1: DRAW i8,j9-j8, 1
280 DRAW i8,j8, 1
285 :
450 mx = (i9-i8)/(x9-x8)
460 my = (j9-j8)/(y9-y8)
480 :
510 FOR v=v8 TO v9 STEP (v9-v8)/20
520 :   FOR u=u8 TO u9 STEP (u9-u8)/20
530 :
540 :     w = EXP(-v*v) * COS(u)
545 :
550 :     x = u + v/SQR(8)
555 :     y = w + v/SQR(8)
```

```
560 :      i2% = i8 + mx * (x-x8)
570 :      j2% = j8 + my * (y-y8)
575 :
610 :      IF u<>u8
           THEN MOVE i1%, j1%: DRAW i2%, j2%, 1
615 :      i1% = i2%: j1% = j2%
700 :   NEXT u
710 NEXT v
715 :
720 a$=INKEY$: IF a$="" THEN 720
```

15.4 Hidden – Line – Algorithmus

Das im letzten Kapitel beschriebene Programm zum Zeichnen von Funktionen w=w(u,v) wollen wir nun durch einen Algorithmus ergänzen, der die unsichtbaren Linien unterdrückt. Zur Erklärung betrachten wir das Relief der Abb. 22.

Die Linie 1 wird von links nach rechts gezeichnet, d. h. für einen festgehaltenen v-Wert läuft u von u8 bis u9. Nach der Berechnung von w = cos(u)*exp(−v*v) wird (u,v,w) in (x,y) und danach (x,y) in den physikalischen Bildpunkt (i2%,j2%) umgerechnet. Dieser alte Bildpunkt wird in i1%=i2%,j1%=j2% gespeichert. Nachdem u erhöht und der neue Bildschirmpunkt i2%,j2% berechnet ist, wird eine Linie von (i1%,j1%) nach (i2%,j2%) gezeichnet (s. Zeile 650 des folgenden Programmes). Auf diese Weise wird die Linie 1 gezeichnet. Die Abb. 22 zeigt, daß die Linie 2 teilweise durch die Linie 1 verdeckt wird. Da die Linie 2 nur bei „Sichtbarkeit" gezeichnet werden soll, also die unsichtbaren Teilstücke wegfallen, speichern wir die Ordinatenwerte aller Linienendpunkte in ein Feld. Für jeden Feldindex k = 0,1,2,. . .,639 enthält dieses Feld h9%(k), den höchsten Ordinatenwert aller bisher gezeichneten Linien. Der Feldindex k entspricht dem x-Wert auf der x-Achse. In dem Speicher h9%(k) ist immer ein Wert zwischen 0 und 399 vorhanden. Diese Speicherinhalte entsprechen den y-Werten der höchsten Begrenzungslinie.

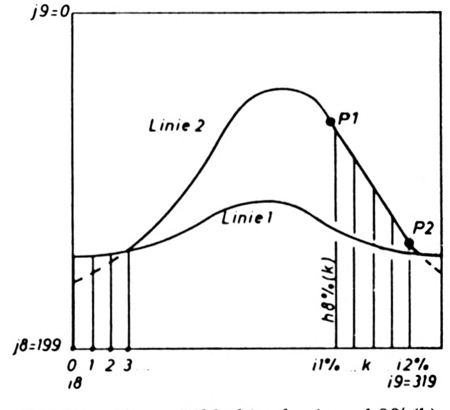


Abb. 22: Oberer Bildschirmhorizont h9%(k)

Eine Linie von P1 nach P2 ist nur dann ganz sichtbar, wenn der linke Linienendpunkt P1 (i1%,j1%) oberhalb von h9%(i1%) und auch der rechte Linienendpunkt p2 (i2%,j2%) oberhalb von h9%(i2%) liegt. Nur in diesem Fall wird die Linie von P1 nach P2 gezeichnet und der Teilbereich h9%(il%). . .h9%(k). . .h9%(i2%) des Bildschirmhorizontes entspre-

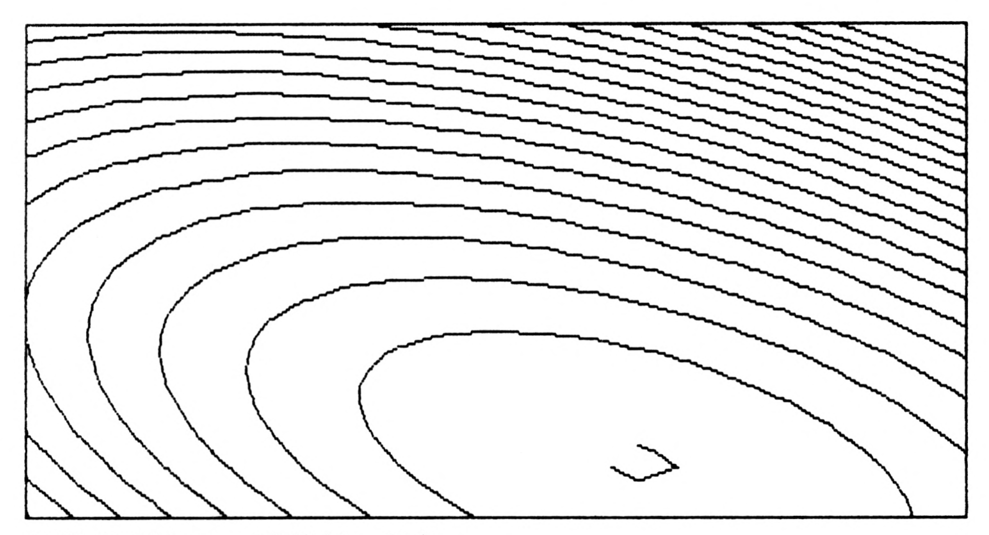
chend aktualisiert. Für k = i1%,i1%+1,...,i2% ist gemäß der Geradengleichung

```
h9%(k) = j1% + (k-i1%)*(j2%-j1%)/(i2%-i1%)
```

der Horizont mit den höheren Horizontwerten zu besetzen (siehe Zeile 620). Die Variable p (s. Zeilen 605,620) gibt an, ob beide Linienendpunkte sichtbar sind. Sie beinhaltet den Wert Null, wenn mindestens ein Linienendpunkt verdeckt wird. Ist p=0, so werden die DRAW-Befehle (s. Zeile 645) übersprungen.

Analog dem oberen Horizont h9%(k) wird mit dem unteren Horizont h8%(k) gearbeitet. Eine Linie wird nur dann gezeichnet (p gesetzt), wenn beide Endpunkte des Linienstückes oberhalb von h9%(k) oder beide Endpunkte unterhalb von h8%(k) liegen. Alle Elemente von h8%(k) werden in Zeile 470 mit 399 vorbesetzt. Die erste Linie auf dem Bildschirm liegt über dem unteren Bildschirmrand (0) und überschreibt dadurch die h9%(k)-Werte (Zeilen 615,620). Ebenso überschreibt die erste Linie den h8%(k)-Horizont (Zeilen 625,630).

Eine Fläche wird besonders anschaulich, wenn begrenzende Linien in u- und v-Richtung gezeichnet werden. Um auch in v-Richtung zeichnen zu können, werden beim Zeichnen in u-Richtung alle berechneten Linienendpunkte i2%,j2% in den Feldern i2%(i) und j2%(i) aufgehoben (Zeile 670). Zudem wird im p%(i)-Feld gespeichert (Zeile 680), ob diese Linie sichtbar war. In der Zeile 660 wird bei Sichtbarkeit der alten Linie der u,w-Ebene eine Gerade vom neuberechneten Punkt (i2%,j2%) in der parallelen u,w-Ebene zum alten Punkt (i2%(i),j2%(i)) gezeichnet. Neutralisiert man die Zeile 660 durch ein eingefügtes „REM“ hinter der Zeilennummer, so ergeben sich nur Linien in u-Richtung (s. Grafik 62).



Grafik 62: Linien ausschließlich in u*-Richtung*

Wir besprechen nun der Reihe nach die Programmzeilen zum Zeichnen einer Funktion w = w(u,v) über einem rechteckigen u,v-Grundgebiet.

Die aktuelle Funktion ist in dem Unterprogramm Zeile 160 bis 180 enthalten. In der Zeile 210 sind die u-Grenzen u8,u9 und v-Grenzen v8,v9 sowie die Anzahl nu der u-Intervalle und die Anzahl nv der v-Intervalle enthalten. Die Zeile 220 legt den Bildschirmausschnitt fest. Es wurde jeweils 0.5 addiert, um eine Rundung der Werte zu erreichen. W1 gibt den Winkel (in Grad) zwischen der y-Achse und der u-Achse (Zeile 230) und w2 den Winkel (in Grad) zwischen der y-Achse und der v-Achse an. Der Wert w0 dient zum Umrechnen der Winkel in das Bogenmaß. In den Zeilen 240,250 werden die trigonometrischen Funktionen berechnet, die als Koeffizienten bei der Umrechnung eines Raumpunktes (u,v,w) in die Zeichenkoordinaten

```
x = -u*sin(w1) + v*sin(w2)
y = +u*cos(w1) + v*cos(w2) + w
```

auftreten (s. 15.1).

In der Zeile 280 können die kleinsten (x8,y8) und größten (x9,y9) Werte der Zeichenebene eingetragen werden. Gewöhnlich liegt x8 in derselben Größenordnung wie u8 und x9 in derselben wie u9. Dagegen wird y8 und y9 stark vom Extremwert w der Funktion w = w(u,v) beeinflußt. Das Programmstück

```
290 PRINT " Berechnen der Extremwerte  ";:
    PRINT "xmin, xmax, ymin, ymax  ";
300 PRINT "der Zeichenebene:":
    PRINT " v = ";
320 :
330 FOR v=-2 TO 2 STEP 0.4
335 :   PRINT v;
340 :   FOR u=-2 TO 2 STEP 0.4
345 :
350 :     w = EXP(-v*v)*COS(u): REM GOSUB 160
360 :     x = -u*s1 + v*s2
370 :     y =  u*c1 + v*c2 + w
375 :
380 :     IF x<x8 THEN x8=x
390 :     IF x>x9 THEN x9=x
400 :     IF y<y8 THEN y8=y
410 :     IF y>y9 THEN y9=y
420 :   NEXT u
425 NEXT v: PRINT: PRINT
430 :
440 PRINT "280 x8= ";x8;":x9= ";x9;":y8= ";y8;
    ":y9= ";y9;":goto 450"
```

berechnet die unbekannten Extremwerte x8,x9,y8,y9 für die Maßstabsfaktoren mx,my der Zeichenebene. Dieses Programmstück ist in etwas geänderter Form in dem Programm „3-dim-Darstellung von w = w(u,v)" enthalten. Mit dem berechneten Funktionswert w (Zeile 350) werden die x und y-Werte berechnet und das jeweils kleinste und größte x und y (Zeilen 380,390,400,410) in x8,y8 und x9,y9 gespeichert und durch Zeile 440 angezeigt. Indem die angezeigte Ausgabe mit der Zeilennummer 280 mit dem COPY-CURSOR übernommen und die ENTER-Taste betätigt wird, wird die inaktivierte Zeile 280 durch die aktuelle Zeile ersetzt. Wenn x8 kleiner gewählt wird, so verschiebt sich die Zeichenfigur nach rechts.

Entsprechendes gilt für x9,y8,y9. Die Umrechnung der Zeichenkoordinaten i2%,j2% geschieht mit Hilfe der Maßstabsfaktoren mx,my (Zeile 450,460).

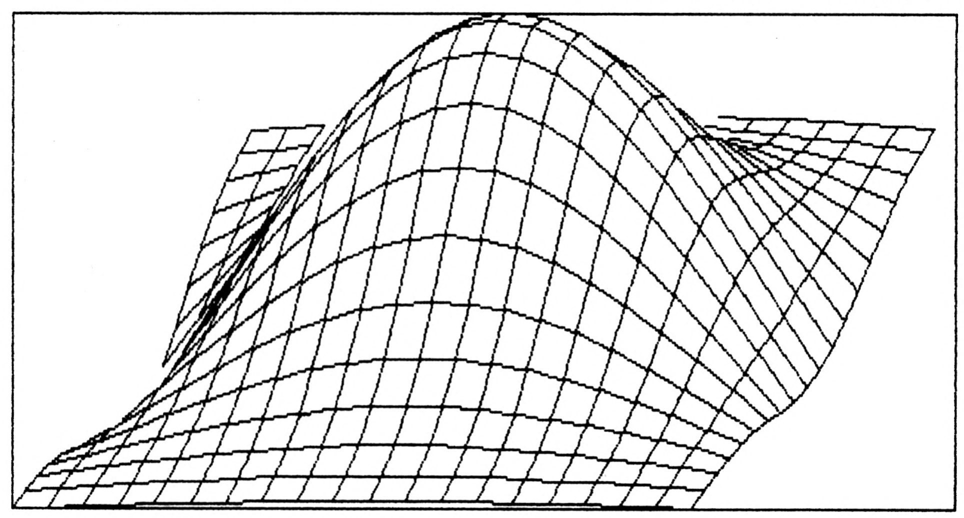
In der Zeile 470 wird der untere Horizont h8%(k) mit 399 vorbesetzt. Der obere Horizont h9%(k) beinhaltet den Anfangswert 0. Durch die Zeile 500 wird der hochauflösende Bildschirm eingeschaltet. Die beiden geschachtelten Schleifen (510,540) beinhalten die vollständige Berechnung der physikalischen Bildkoordinaten i2%,j2% und den HIDDEN-LINE-Algorithmus. Entsprechend den Gleichungen

```
x = -u * s1 + v * s2
y = +u * c1 + v * c2 + w
```

und

```
i2% = i8 + mx * (x-x8)
j2% = j8 + my * (y-y8)
```

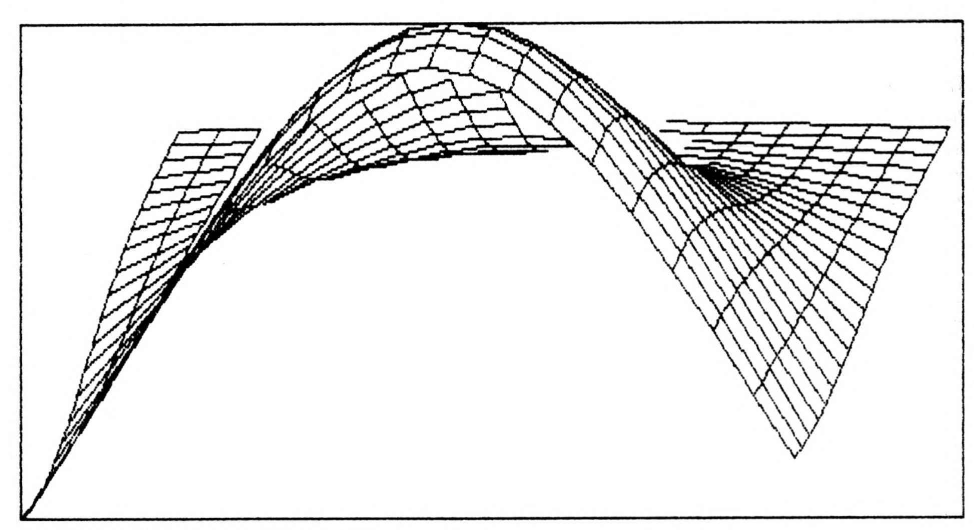
ist ein Teil der Berechnungen aus der innersten Schleife (Zeile 520) herausgenommen worden. Liegt ein berechneter Punkt i2%,j2% außerhalb des Bildschirmes (i8=0,i9=639,j8=0,j9=399), so wird (i2%,j2%) als Randpunkt des Bildschirmes angenommen (Zeilen 580,585,590,595). In den Zeilen 615,620 wird der Horizont h9%(k) und



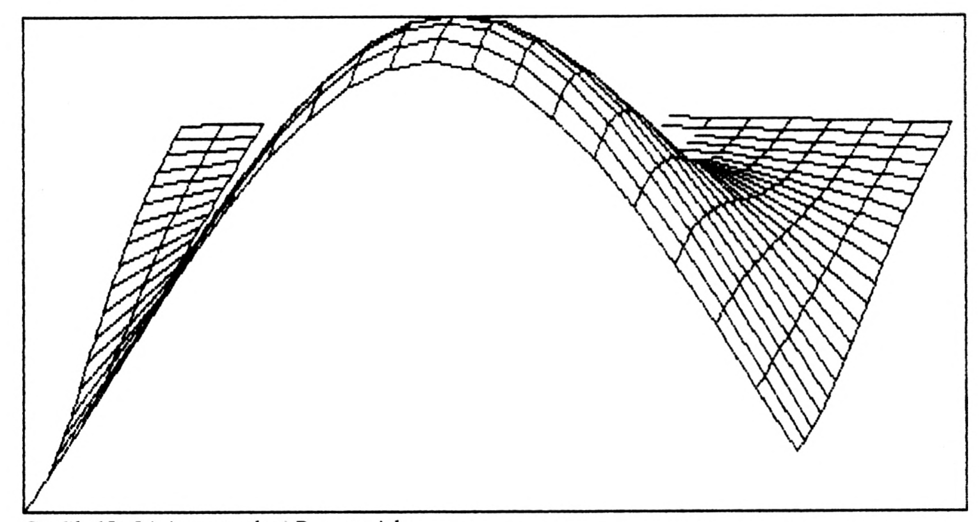
Grafik 63: Liniennetz mit HIDDEN-LINE-Algorithmus zur grafischen Darstellung der Funktion
w = exp(−v∗v) ∗ cos(u)

in den Zeilen 625,630 der Horizont h8%(k) besetzt. Dabei wird auch die „Sichtbarkeitsflagge“ p gesetzt. Abhängig von p wird eine Linie von p1(i1%,j1%) nach P2(i2%,j2%) gezeichnet (Zeile 650). Diese Linie verläuft in einer parallel zur w,u-Ebene liegenden Ebene.

Die Zeile 660 veranlaßt bei vorliegender Sichtbarkeit das Zeichnen einer Linie in einer zur w,v-Ebene parallelen Ebene. Dadurch erhalten wir ein rechteckförmiges Netz (Grafik 63). Die Netzdichte wird durch nu,nv (Zeile 210) festgelegt. Wird die Zeile 660 durch ein eingefügtes „REM“ nicht bearbeitet, so ergibt sich die Grafik 62. Die Grafik 64 erhält man, wenn in der Zeile 210 die v-Grenze v8=−0.2 gesetzt wird. Der Horizont h8%(k) bewirkt, daß wir



Grafik 64: Liniennetz mit Untersicht



Grafik 65: Liniennetz ohne Druntersicht

das Innere der Figur sehen. Wird die Zeile 630 durch ein eingeschobenes „REM“ inaktiviert, so ist der Horizont h8%(k) nicht mehr vorhanden, und wir erhalten die Grafik 65.

```
100 REM---------------------------------------
105 REM-- 3-dim-Darstellung von w = w(u,v)   --
107 REM---------------------------------------
110 REM-- Unterprog fuer w   Zeilen 160-180 --
115 REM-- min-max-Berechnung Zeilen 290-440 --
120 REM-- Linien zeichnen    Zeile  650     --
125 REM-- Hidden-Line-Algorithmus   600-680 --
130 :
140 PRINT " Besetzen der Startwerte :   ";
150 GOTO 200
155 :
160 REM--- Unterprogramm fuer w = w(u,v) ---
165 :
170 w=EXP(-v*v)*COS(u)
175 :
180 RETURN
190 :
195 :
200 DIM i2%(100), j2%(100), p%(100),
        h8%(639), h9%(639)
210 u8 = -2: u9 = 2: v8 = -2: v9 = 2:
    nu = 20: nv = 20
220 i8 =0.5: i9 =639.5: j8 =0.5: j9 =399.5
230 w1 = 270: w2 = 45: w0 = PI/180
240 s1 = SIN(w1*w0):  s2 = SIN(w2*w0)/2
250 c1 = COS(w1*w0):  c2 = COS(w2*w0)/2
260 PRINT "o.k."
270 :
280 REM--- x8=-3:x9= 3:y8=-1:y9= 1.2: goto 450
285 :
290 PRINT " Berechnen der Extremwerte  ";:
    PRINT "xmin, xmax, ymin, ymax  ";
300 PRINT "der Zeichenebene":
    PRINT " fuer v = ";
310 :
320 x8 =9E+09: x9 =-x8: y8 = x8: y9 =-x8
325 :
330 FOR v=v8 TO v9 STEP (v9-v8)/nv/2
335 :  PRINT v;
```

```
340 :  FOR u=u8 TO u9 STEP (u9-u8)/nu/2
350 :    GOSUB 160
360 :    x = -u*s1 + v*s2
370 :    y =  u*c1 + v*c2 + w
380 :    IF x<x8 THEN x8=x
390 :    IF x>x9 THEN x9=x
400 :    IF y<y8 THEN y8=y
410 :    IF y>y9 THEN y9=y
420 :  NEXT u
425 NEXT v: PRINT: PRINT
430 :
440 PRINT "280 x8= ";x8;":x9= ";x9;":y8= ";y8;
    ":y9= ";y9;":goto 450"
445 :
450 mx = (i9-i8)/(x9-x8)
460 my = (j9-j8)/(y9-y8)
465 :
470 FOR k=0 TO 639: h8%(k) = j9: NEXT
480 :
490 REM- berechnen und zeichnen von w=w(u,v) -
500 MODE 2:BORDER 2:INK 0,2:INK 1,24
504 ti=TIME
505 :
510 FOR v=v8 TO v9 STEP (v9-v8)/nv
520 : hx = v*s2 - x8: hy = v*c2 - y8: i=-1
540 : FOR u=u8 TO u9 STEP (u9-u8)/nu
545 :
550 :  GOSUB 160: REM w=w(u,v) berechnen
560 :  i2% = i8 + mx*(hx-u*s1)
570 :  j2% = j8 + my*(hy+u*c1+w)
580 :  IF i2%<i8 THEN i2%=i8
585 :  IF i2%>i9 THEN i2%=i9
590 :  IF j2%<j8 THEN j2%=j8
595 :  IF j2%>j9 THEN j2%=j9
600 REM--- Horizont h8%(),h9%() besetzen
605 :: h=0: p=0
610 :: IF i1%<>i2% THEN h=(j2%-j1%)/(i2%-i1%)
612 :
615 :: IF j1%<h9%(i1%) OR j2%<h9%(i2%)
       THEN 625
620 :: FOR k=i1% TO i2%:
       h9%(k) = j1% + h*(k-i1%): NEXT: p=1
```

```
622 :
625 :: IF j1%>h8%(i1%) OR j2%>h8%(i2%)
       THEN 645
630 :: FOR k=i1% TO i2%:
       h8%(k) = j1% + h*(k-i1%): NEXT: p=2
635 :
640 REM--- sichtbare Linien zeichnen ---
645 :: i=i+1: IF p=0 THEN 670
650 :: IF u<>u8 THEN
       MOVE i1%, j1%: DRAW i2%, j2%, 1
660 :: IF p%(i) THEN
       MOVE i2%, j2%: DRAW i2%(i), j2%(i), 1
670 :: i2%(i) = i2%: j2%(i) = j2%
680 :: i1% = i2%: j1% = j2%: p%(i) = p
690 :
700 :  NEXT u
710 NEXT v
910 :
920 PRINT "Zeichenzeit ="; (TIME-ti)/18000;
     " min"
930 a$=INKEY$: IF a$="" THEN 930
```

Das angegebene Programm kann auch zum Zeichnen von zusammengesetzten Funktionen verwendet werden. Diese Funktionen w = w(u,v) sind in dem Unterprogramm ab der Zeile 160 zu definieren. Geben wir zum Beispiel

```
170 w = (1+SIN(u)) * (1+SIN(v))
175 IF w>1.8 AND v<1 AND ABS(u)<3 THEN w = 1.8
180 RETURN
```

ein, so erhalten wir mit der Zeile

```
210 u8 =-5: u9 = 5: v8 =-5: v9 = 5: nu =40: nv =40
```

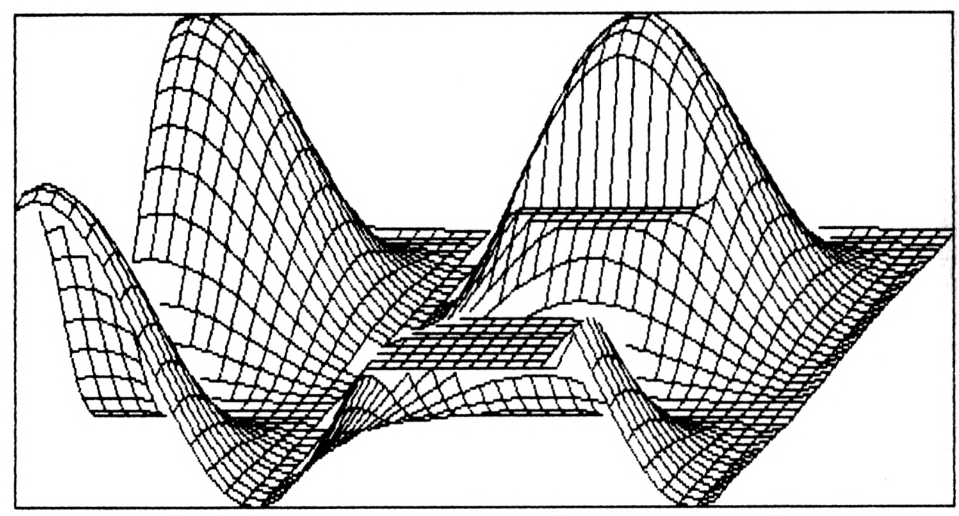
die Grafik 66.

Wir wollen nun mit dem gleichen Programm die Oberfläche einer Figur zeichnen lassen, die sich durch 2 kreuzende Zylinder ergibt. die Zylinderachsen liegen in u- bzw. v-Richtung. Die Zylinderradien seien r1=6,r2=5. Gemäß der kartesischen Kreisgleichung ist für ein konstantes v die Höhe h1 über der u,v-Ebene durch

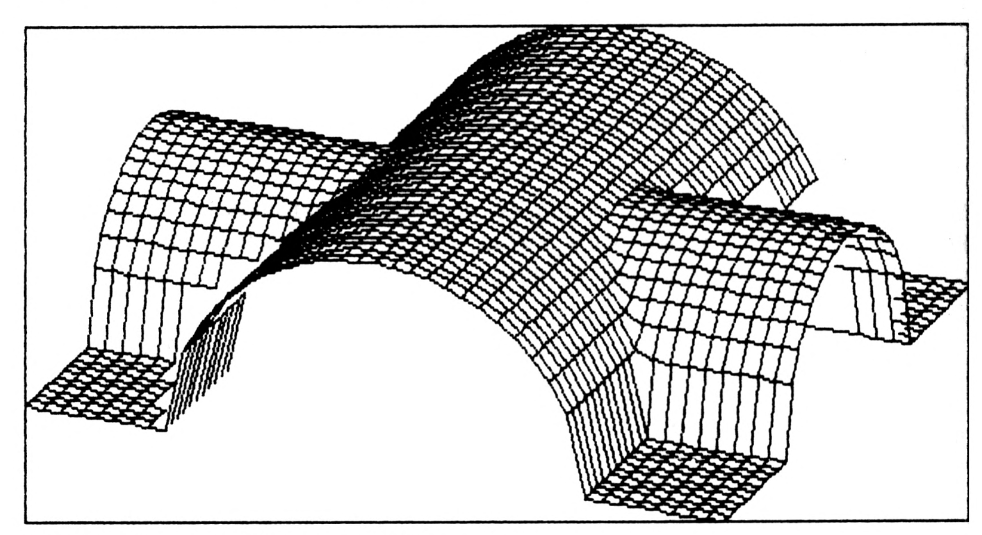
```
h1 = SQR(r1*r1 - u*u)
```

gegeben. Entsprechend gilt

```
h2 = SQR(r2*r2 - v*v).
```



Grafik 66: 3D-Darstellung einer „abgeschnittenen" Funktion



Grafik 67: Zwei geschnittene Halbzylinder

Da wir den Aufriß darstellen wollen, nehmen wir für w den größeren der beiden Werte h1,h2. Für das Unterprogramm zur Berechnung von w ergibt sich somit

```
160 REM--- Unterprogramm für w = w(u,v) ---
162 w = 0: h1 = 0: h2 = 0
166 IF ABS(u)<=6 THEN h1 = SQR(36 - u*u)
168 IF ABS(v)<=5 THEN h2 = SQR(25 - v*v)
172 IF h1>w THEN w = h1
```

```
176 IF h2>w THEN w = h2
180 RETURN.
```

Die Grafik entstand mit der Zeile

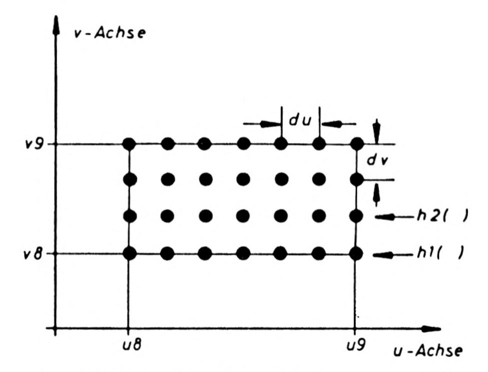
```
210 u8 =-10: u9 =10: v8 =-10: v9 =10: nu =40: nv =40.
```

Weiterhin wurde die Figur durch w1 = 260° (s. Zeile 230) etwas gedreht. Ohne Min-Max-Berechnung benötigt das Programm 6 Minuten zum Zeichnen der Grafik 67. Der Zeitbedarf hängt wesentlich von der gewählten engmaschigen Einteilung (nu=40) ab.

15.5 Niveaulinien

Niveaulinien werden in unterschiedlichen Disziplinen zur Kennzeichnung von Höhenunterschieden verwendet. In der Geodäsie verwendet man Niveaulinien zum Markieren der Geländehöhen. Kegelförmige Berge ergeben ellipsenförmige Höhenlinien. Würde man im Gelände auf einer solchen Niveaulinie entlanggehen, so würde man keinen Höhenunterschied bemerken.

Liegen auf einer Landkarte die Höhenlinien dicht beieinander, ist ein steiler Geländeanstieg vorhanden. Weit auseinanderliegende Niveaulinien deuten auf ein ebenes Gelände hin. Die Richtung des steilsten Geländeanstieges nennt man auch Gradientenrichtung. Die Niveaulinien sind immer senkrecht zum Gradienten. In der Physik (Vektoranalysis) werden Niveaulinien zur grafischen Darstellung von skalaren Ortsfunktionen (Potentiale) verwendet.



Ab. 23: Rechteckiges Grundgebiet zur Berechnung der Stützhöhen

Wir wollen im folgenden die gegebene Funktion w=w(u,v) mit Hilfe von Höhen linien grafisch darstellen. Der Funktion w=w(u,v) entspricht eine Fläche über der u,v-Ebene.

Die gesuchte grafische Niveaulinien-Darstellung entspricht der Projektion aller gleichen Höhen in die u,v-Ebene. Zum Entwickeln eines Programmes, das diese Niveaulinien zeichnet, betrachten wir zunächst ein rechteckiges Grundgebiet, in dem die Höhenlinien gezeichnet werden sollen.

Dieses Grundgebiet liegt in der u,v-Ebene (s. Abb. 23) und wird begrenzt durch u8, u9, v8, v9. Alle u-Werte liegen zwischen u8 und u9 und alle v-Werte zwischen v8 und v9. In den Programmzeilen 880–910 werden die Stützhöhen für die erste u-Reihe (v=v8; s. Abb. 28) von u=u8 bis u=u9 mit der Schrittweite du berechnet. Diese Höhen werden in dem Feld h1() gespeichert. Die Höhen der nächsten u-Reihe (v=v8+dv) werden berechnet (Zeilen 950–970) und in dem Feld h2() gespeichert.

Wir betrachten nun die Abb. 24. Die Entfernung vom Punkt A bis Punkt B entspricht du und die Entfernung von B nach C dem Wert dv. Hierbei entspricht du (Zeile 510) einer Unterteilung des u8, u9-Bereiches. Die vier Stützhöhen h1(i), h1(i+1), h2(i), h2(i+1) in den Ecken eines kleinen du,dv-Grundgebietes seien berechnet und somit bekannt. Die obere Begrenzungsfläche entspricht näherungsweise einer Ebene. Die zu zeichnende Linie liegt in der u,v-Ebene und ist in der Abb. 24 strichpunktiert gezeichnet. Wir ermitteln nun die Koordinaten der Punkte P0 und Pk.

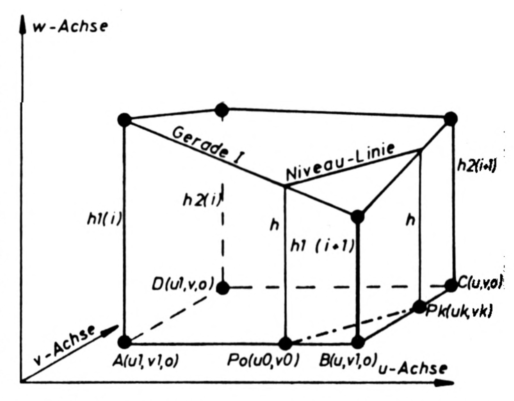


Abb. 24: Ermittlung von P0,Pk *für die strichpunktierte Niveaulinie mit der Höhe* h

Sind diese Punkte bekannt, so kann die strichpunktierte Gerade gezeichnet werden. Betrachten wir die senkrechte Ebene über der Strecke A—B (s. Abb. 24), so ergibt sich mit dem Strahlensatz oder der Parameterdarstellung der Geraden I der Punkt P0 gemäß:

```
t = (h-h1(i))/(h1(i+1)-h1(i))
u0 = u1 + t*(u-u1)
v0 = v1
```

Der Punkt P0 liegt nur dann zwischen A und B, wenn der t-Wert zwischen 0 und 1 liegt.

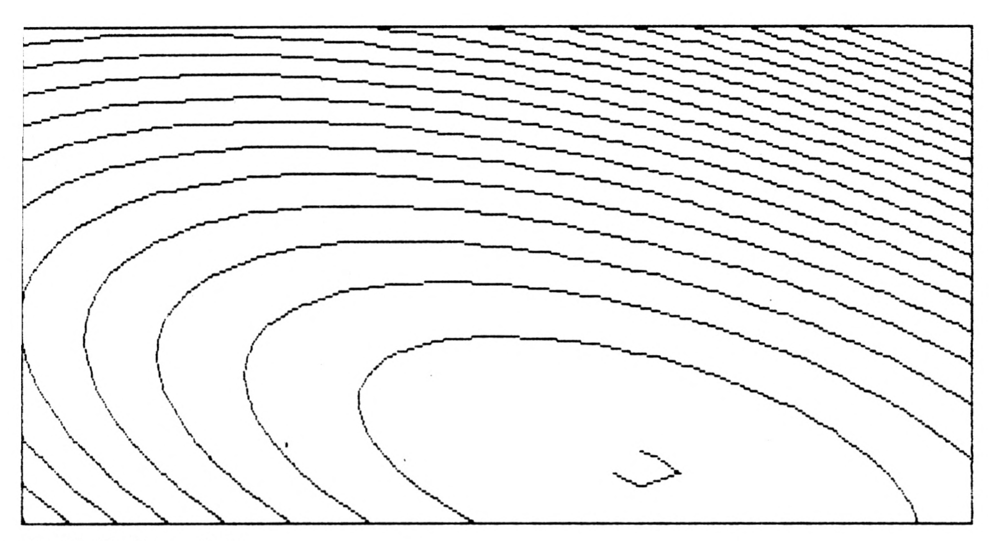
Betrachten wir die senkrechte Ebene über der Strecke B—C, so ergibt sich der Punkt Pk(uk,vk) gemäß:

```
t = (h-h1(i+1))/(h2(i+1)-h1(i+1))
uk = u
vk = v1 + t*(v-v1).
```

Der Punkt Pk liegt zwischen B und C, wenn t zwischen 0 und 1 liegt.

In ähnlicher Weise können wir alle 4 senkrechten Bezugsflächen des du,dv-Grundgebietes betrachten. Zu einer vorgegebenen Höhe h gibt es bei einer geneigten Ebene über dem du,dv-Grundgebiet gewöhnlich zwei Schnittpunkte mit den senkrechten Begrenzungsflächen. In den Zeilen 1070 bis 1140 werden diese beiden Schnittpunkte für die Höhe h ermittelt, und nach der Umrechnung für die Bildschirmkoordinaten (Zeilen 1160—1190) wird die Höhenlinie gezeichnet.

In dem du,dv-Grundgebiet werden jeweils alle Niveaulinien (Geraden) gezeichnet (Zeile 1020). Das du,dv-Gebiet wird in u-Richtung (von links nach rechts) durchgeschoben (Zeile 1010). Die bereits errechneten Höhen h2() werden in das Feld h1() umgespeichert (Zeilen 1260—1280), und der v-Wert wird erhöht (Zeile 930).



Grafik 68: Niveaulinien

Die Grafik 68 erhalten wir, wenn wir in der Zeile 230 die Funktion

```
230 w = u*(u+v) + v*(v+1)
```

eingeben. Beim Programmlauf verwenden wir die voreingestellten Werte (Zeile 300). Die vielen PRINT-Anweisungen erklären dem Anwender die notwendigen Programmeingaben.

```
100 REM-------------------------------------
105 REM- Zeichnen von Hoehen-schicht-linien -
110 REM- fuer die Funktion  w = w(u,v)      -
115 REM-------------------------------------
125 :
130 MODE 2:PRINT
140 PRINT"  Bitte geben Sie die Funktion ein."
150 PRINT"  Der Programmlauf wird dann mit"
160 PRINT:PRINT"     run 300    <enter>  "
170 PRINT
180 PRINT"                      fortgesetzt."
190 PRINT:PRINT
200 LIST 220-240
210 :
220 REM--- Unterprog. fuer w=w(u,v) ---
230 w = u*(u+v) + v*(v+1)
240 RETURN
250 :
```

```
300 u8 =-0.9: u9 = 1: v8 = -0.9: v9 = 1.2
310 nu = 20: nv = 30: nw = 20
320 i8 =0.5: i9 =639.5: j8= 0.5: j9 =399.5
330 PRINT"  Min u-Wert     u min =  ";u8
340 PRINT"  Max u-Wert     u max =  ";u9
350 PRINT"  Anz.von u-Intervall =  ";nu
360 PRINT"  Min v-Wert     v min =  ";v8
370 PRINT"  Max v-Wert     v max =  ";v9
380 PRINT"  Anz.von v-Intervall =  ";nv
390 PRINT
400 INPUT"Vorbesetzte Werte aendern ";a$
410 IF LOWER$(a$)<>"j" THEN 510
420 :
430 PRINT:PRINT
440 INPUT"  Min u-Wert     u min =  ";u8
450 INPUT"  Max u-Wert     u max =  ";u9
460 INPUT"  Anz.von u-Intervall =  ";nu
470 INPUT"  Min v-Wert     v min =  ";v8
480 INPUT"  Max v-Wert     v max =  ";v9
490 INPUT"  Anz.von v-Intervall =  ";nv
500 :
510 du = (u9-u8)/nu: dv = (v9-v8)/nv
520 :
530 PRINT:PRINT
540 PRINT:
PRINT"Berechnen der Extremwerte w min, w max"
550 PRINT:PRINT
560 w8=9E+09: w9=-9E+09
570 :
580 FOR v=v8 TO v9 STEP dv*2
590 :  FOR u=u8 TO u9 STEP du*2
600 :     GOSUB 220
610 :     IF w>w9 THEN w9=w
620 :     IF w<w8 THEN w8=w
630 :  NEXT u
640 NEXT v
650 :
660 PRINT"  Min Hoehe     w min =    ";w8
670 PRINT"  Max Hoehe     w max =    ";w9
680 PRINT"  Anzahl der Niveaus =    ";nw
690 PRINT
```

```
700 INPUT"Vorbesetzte Werte aendern ";a$
710 IF LOWER$(a$)<>"j" THEN 770
720 :
730 PRINT:PRINT
740 INPUT"  Min Hoehe    w min =   ";w8
750 INPUT"  Max Hoehe    w max =   ";w9
760 INPUT"  Anzahl der Niveaus =   ";nw
770 DIM h1(nu), h2(nu)
780 :
790 dw = (w9-w8)/nw
800 mx = (i9-i8)/(u9-u8)
810 my = (j9-j8)/(v9-v8)
820 :
830 REM- Zeichnen der Hoehen-schicht-linien -
840 BORDER 3:INK 0,3:INK 1,24:CLG 0
850 MOVE 0,0:DRAW 639,0, 1:DRAW 639,399, 1
860 DRAW 0,399, 1:DRAW 0,0, 1:     tt = TIME
870 :
880 :     v = v8: k = -1
890 :     FOR u=u8 TO u9+du/2 STEP du
900 :        GOSUB 220: k=k+1: h1(k) = w
910 :     NEXT u
920 :
930 FOR v=v8+dv TO v9+dv/2 STEP dv: k = -1
940 :
950 :     FOR u=u8 TO u9+du/2 STEP du
960 :        GOSUB 220: k=k+1: h2(k) = w
970 :     NEXT u
980 :
990 : v1 = v - dv
1000 :
1010 : FOR i=0 TO nu-1:  u1=u8+i*du: u=u1+du
1020 :    FOR h=w8 TO w9 STEP dw: k = -1
1030 :
1040 :    IF h1(i)<h AND h1(i+1)<h AND h2(i)<h
            AND h2(i+1)<h THEN 1220
1050 :    IF h1(i)>h AND h1(i+1)>h AND h2(i)>h
            AND h2(i+1)>h THEN 1220
1060 :
1070 :    t=h2(i+1)-h1(i+1):IF t=0 THEN 1090
1080 :    t=(h-h1(i+1))/t:  IF t>=0 AND t<=1
          THEN k=k+1: v(k)=v1+t*(v-v1): u(k)=u
```

```
1090 :    t=h2(i+1)-h2(i):   IF t=0 THEN 1110
1100 :    t=(h-h2(i))/t:     IF t>=0 AND t<=1
          THEN k=k+1: u(k)=u1+t*(u-u1): v(k)=v
1110 :    t=h2(i) - h1(i):   IF t=0 THEN 1130
1120 :    t=(h-h1(i))/t:     IF t>=0 AND t<=1
          THEN k=k+1: v(k)=v1+t*(v-v1): u(k)=u1
1130 :    t=h1(i+1)-h1(i):   IF t=0 THEN 1150
1140 :    t=(h-h1(i))/t:     IF t>=0 AND t<=1
          THEN k=k+1: u(k)=u1+t*(u-u1): v(k)=v1
1150 :    IF k<>1 THEN 1220
1160 :        i1% = i8 + mx*(u(0)-u8)
1170 :        j1% = j8 + my*(v(0)-v8)
1180 :        i2% = i8 + mx*(u(1)-u8)
1190 :        j2% = j8 + my*(v(1)-v8)
1200 :        MOVE i1%,j1%: DRAW i2%,j2%, 1
1210 :
1220 :    NEXT h
1230 :
1240 :  NEXT i
1250 :
1260 :      FOR k=0 TO nu
1270 :         h1(k) = h2(k)
1280 :      NEXT k
1290 :
1300 NEXT v
1310 :
1315 LOCATE 1,1
1320 PRINT " Zeit="; (TIME-tt)/18000;" min"
1330 a$=INKEY$: IF a$="" THEN 1330
```

Eine andere Methode zum Darstellen von Punkten mit gleicher Höhe soll nun erläutert werden. Wir gehen davon aus, daß die Funktion z=z(x,y) gegeben ist. In der Zeile 720 wird z. B. die Funktion

```
z = x*(x+y) + y*(y+1)
```

verwendet. Natürlich ist die beschriebene Methode auf jede andere Funktion z=z(x,y) übertragbar. Wir wollen die Grafik farbig gestalten. Deshalb wird durch den MODE-Befehl in Zeile 610 der Normalbildschirm eingeschaltet und danach die Rahmenfarbe Hellblau (2) aktiviert. In diesem Standard-Bildschirmmodus ist es möglich, vier der 27 verschiedenen Farben gleichzeitig darzustellen. Diese vier Farben werden in der Zeile 615 den Farbstiften 0 bis 3 zugeordnet. Es sind dies die Farben Blau, Weiß, Gelb und Grün. Durch die Zeilen 620,625 wird dann ein weißer Bildrahmen gezeichnet. In der Zeile 630 hinterlegen wir die

x-Grenzen (x8,x9) und die y-Grenzen (y8,y9) für die zu zeichnende Funktion z=(x,y), die in der Zeile 720 eingegeben wird. Die Werte z8,z9 in der Zeile 640 stellen die minimalen und maximalen z-Werte der gegebenen Funktion z=z(x,y) dar, für die Höhenschichten zu zeichnen sind. Die Anzahl nz der Höhenschichten ist in Zeile 645 hinterlegt.

```
600 REM--- farbige Hoehenschichten ---
610 MODE 1:BORDER 2
615 INK 0,1:INK 1,26:INK 2,12:INK 3,22
620 MOVE 0,0: DRAW 639,0, 1:DRAW 639,399, 1
625 DRAW 0,399, 1:DRAW 0,0, 1
627 :
630 x8 = -2: x9 =  2: y8 = -2: y9 = 2
640 z8 = -1: z9 = 13
645 nz = 30
647 :
650 bi = 2: i8 =  2: i9 =637
655 bj = 2: j8 =  2: j9 =397
660 bz = (z9-z8)/nz: z6=9E+09: z7=-9E+09
665 mx = (i9-i8)/(x9-x8)
670 my = (j9-j8)/(y9-y8)
675 :
680 FOR j=j8 TO j9 STEP bj
690 :  y = y8 + (j-j8)/my
700 :  FOR i=i8 TO i9 STEP bi
710 :     x = x8 + (i-i8)/mx
720 :     z = x*(x+y)+y*(y+1)
740 :     REM--- p = (z-z8)/bz MOD 4
750 :     p = (z-z8)/bz MOD 2
760 :     PLOT i,j, p
770 :     IF z>z7 THEN z7=z
780 :     IF z<z6 THEN z6=z
790 :  NEXT i
800 NEXT j
810 :
860 a$=INKEY$: IF a$="" THEN 860
865 :
890 PRINT: PRINT: PRINT
900 PRINT " Die extremen Hoehen sind:"
910 PRINT " 630 z8 ="; z6;": z9 ="; z7
```

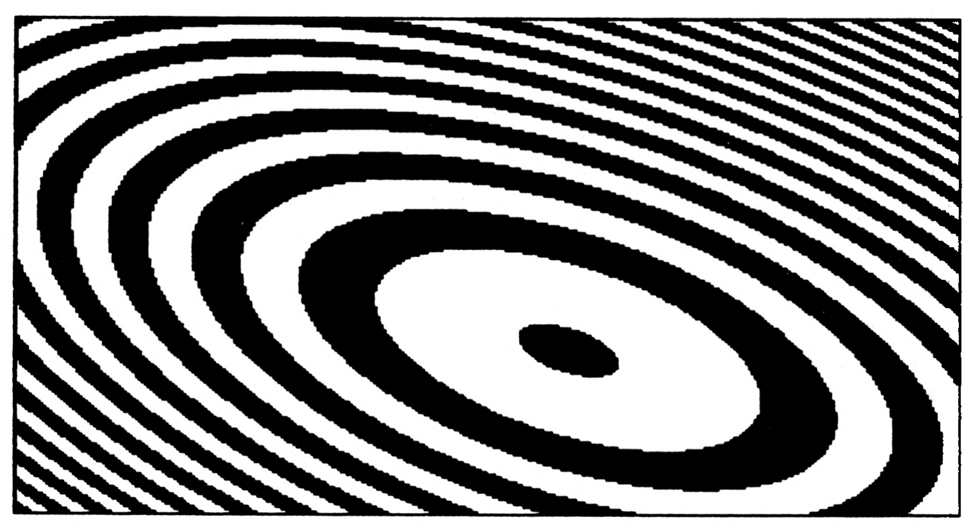
In den Zeilen 650,655 werden die Grenzen i8,i9,j8,j9 für die Bildschirmpixels hinterlegt. Die Werte bi bzw. bj legen die Schrittweite in Bildschirmpixels in x- bzw. y-Richtung fest.

Zum Testen des Programmes sollte bi,bj nicht zu klein gewählt werden (z. B. bi=10,bj=10). Für die endgültige Erstellung der Zeichnung wählen wir bi=2,bj=2, da im normalauflösenden Bildschirmmodus in x- und y-Richtung jeweils zwei benachbarte Pixel zu einem zusammengefaßt werden. Eine Schrittweite von 2 Bildschirmpixels ist also völlig ausreichend. In den Zeilen 665,670 werden die Maßstabsfaktoren mx,my ermittelt. Der Wert bz (Zeile 660) entspricht der Dicke einer z-Schicht. Ist bi=2 und bj=2, so werden durch die beiden Schleifen (Zeilen 680,700) alle Bildschirmpixels von „links nach rechts" und von „unten nach oben" durchlaufen.

Im Kapitel 3.1 werden die Skalengleichungen

```
i = i8 + mx*(x-x8)
j = j8 + my*(y-y8)
```

zur Umrechnung von x,y-Werten in Bildschirmpixels i,j abgeleitet. Weil die Bildschirmwerte i,j durch die Schleifen 680,700 vorgegeben sind, stellen wir diese Gleichungen nach x,y um. Die Berechnung der zu i,j gehörenden x,y-Werte erfolgt in den Zeilen 690,710. Mit diesen x,y-Werten wird nun der Funktionswert z=z(x,y) berechnet (Zeile 720). Durch den Befehl PLOT i,j, 1 kann ein Pixel gesetzt und durch PLOT i,j, 0 gelöscht werden.



Grafik 69: Farbige Darstellung der Höhenschichten einer Funktion z=z(x,y)

Der Befehl w MOD 2 liefert lediglich die Werte 0 oder 1 für positive w-Werte. Durch die Zeile 740 ermitteln wir den Wert p=0,1,2 oder 3 und verwenden diesen Wert als Farbstift. Hierbei entspricht p=0 wegen INK 0,1 der Farbe Blau und p=1,2,3 wegen INK 1,26:INK 2,12:INK 3,22 den Farben Weiß, Gelb, Grün. Soll lediglich eine zweifarbige Grafik erstellt werden, so ist die Zeile 740 zu neutralisieren und die Zeile 750 zu aktivieren. Statt

```
750 p = (z-z8)/bz MOD 2
```

kann auch

```
750 p = (z-z8)/bz AND 1
```

verwendet werden. Wir erhalten dann die Grafik 69. Die Erstellung dieser Grafik (mit bi=2,bj=2) benötigt 31 Minuten. Deshalb sollte beim Testen mit größeren bi,bj-Werten gearbeitet werden.

16. Ausgewählte Anwendungen

16.1 Simulationen

Für die Beschreibung von komplexen physikalisch-technischen Vorgängen ist gewöhnlich ein mathematisches Modell erforderlich, das die wesentlichen Einflußparameter berücksichtigt. In vielen Fällen ist man auf die Simulation von realen Systemen angewiesen, weil die Experimente am Orginalsystem wegen zu hoher Kosten, zu großem Sicherheitsrisiko oder zu großem Zeitbedarf nicht durchführbar sind. Einfache Beispiele für solche Simulationen sind:

- Ballistik einer Gewehrkugel
- Wurf mit Luftwiderstand
- e/m Bestimmung nach Busch
- Elektron im Strahlungsgürtel der Erde
- Compton-Versuch
- Radioaktiver Zerfall
- Verkehrsstau
- Entladen eines Kondensators
- Einschwingvorgänge u. a.

Die mathematische Beschreibung des gewählten Modelles geschieht in vielen Fällen mit Hilfe von Differentialen. Die geschlossene Lösung von schwierigen Differentialgleichungen ist vielleicht unübersichtlich oder nicht auffindbar. Deshalb wird zur numerischen Lösung von Differentialgleichungen der Computer eingesetzt. Zur Vermeidung von unübersichtlichen Zahlenkolonnen bietet sich die grafische Darstellung der numerischen Lösung an. Als Beispiel wollen wir die Differentialgleichungen

```
dx/dt = f(t,x,y)
dy/dt = g(t,x,y)
```

behandeln.

Die vorgegebenen Funktionen f,g hängen von t,x,y ab.

Es sei P0(x0,y0) ein vorgegebener Punkt in der x,y-Ebene und t0 der zu P0 gehörende Parameter, der in der Physik vielfach der Zeit entspricht. Z. B. kann durch die Differentialgleichungen

```
dx/dt = v0 * COS(w0)
dy/dt = v0 * SIN(w0) - g * t
```

der schiefe Wurf (ohne Luftreibung) beschrieben werden. Die Abschußgeschwindigkeit v0 und der Abschußwinkel w0 ist im Punkte P0(x0,y0) zum Zeitpunkt t0 bekannt. Zur Simulation sind diese Differentialgleichungen numerisch zu lösen. Wir verwenden hierzu das folgende Halbschrittverfahren (Extrapolationsmethode).

Die Schrittweite sei dt. Für t = t0 + dt/2 werden die Ableitungen dx/dt = xt,dy/dt = yt berechnet. Mit diesen Ableitungswerten berechnen wir den Punkt P1(x1,y1) gemäß x1 = x0 + dt*xt, y0 + dt*yt. Nun bilden wir t = t0 + 3*dt/2 und berechnen die neuen Ableitungen xt, yt, y1 =. Wir erhalten P2(x2, y2) gemäß x2 = x1 + dt*xt, y2 = y1 + dt*yt. In gleicher Weise fahren wir fort. Wir verwenden jeweils die Ableitungen zwischen 2 aufeinanderfolgenden Punkten. In den Programmzeilen 600 bis 630 werden diese Schritte fortlaufend ausgeführt. Das wiedergegebene Programm kann für viele Anwendungen eingesetzt werden und wird im folgenden beschrieben.

```
100 REM !------------------------------------!
110 REM ! grafische Darstellung der Loesung !
120 REM !   von Differential-Gleichungen    !
130 REM !------------------------------------!
140 REM- Unterprogramm fuer die Ableitungen -
150 REM- xt = f(t,x,y) (xt entspricht dx/dt)-
160 REM- yt = g(t,x,y) (yt entspricht dy/dt)-
170 :
180 w0 = 15 : v0 = 28 : GOTO 230
190 xt = v0*COS(w0)
200 yt = v0*SIN(w0) - 9.81*(t-t0)
210 RETURN
220 :
230 t0 = 0: t9 = 5: dt = 0.1: k = 0
240 x0 =10: y0 =10: xt =  0: yt = 0
250 x8 = 0  : x9 = 100: y8 =   0: y9 = 50
260 :
270 i8 = 7.5: i9 =632.5: j8 = 7.5: j9 =312.5
280 mx = (i9-i8)/(x9-x8): i0 = i8+mx*(x0-x8)
290 my = (j9-j8)/(y9-y8): j0 = j8+my*(y0-y8)
300 :
310 WINDOW #0,1,80,1,5
320 CLS #0
330 INPUT" vorbesetzte Werte aendern (j/n) ";a$
340 IF LOWER$(a$)="j" THEN LOCATE 50,3:
    PRINT"weiter mit  run 350  <ENTER>":
    LOCATE 1,1: LIST 230-250,#0
350 :
360 INPUT" alte Grafik loeschen (j/n) ";a$
370 IF LOWER$(a$)<>"j" THEN 570
380 :
390 MODE 2:BORDER 2:INK 0,1:INK 1,26
400 WINDOW #0,1,80,1,5
410 ORIGIN 0,0,0,639,319,0
```

```
420 :
430 MOVE 0,0: DRAW 639,0,1: DRAW 639,319,1
440 DRAW 0,319,1: DRAW 0,0,1
450 :
460 MOVE i0,j0+12: DRAW i0+12,j0+6, 1
470 DRAW i0+12,j0-6, 1: DRAW i0,j0-12, 1
480 DRAW i0-12,j0-6, 1: DRAW i0-12,j0+6, 1
490 DRAW i0,j0+12, 1
500 :
510 FOR i=i8 TO i9 STEP (i9-i8)/10
520  MOVE i,j8: DRAW i,j9,1: NEXT i
530 FOR j=j8 TO j9 STEP (j9-j8)/10
540  MOVE i8,j: DRAW i9,j,1: NEXT j
550 :
560 REM--- Halbschritt-Verfahren ---
570 DEG
580 x=x0: y=y0: t=t0 + dt/2: GOSUB 190: k=0
590 WHILE t<=t9
600 :   GOSUB 190
610 :   t = t + dt
620 :   x = x + xt*dt
630 :   y = y + yt*dt
640 :   IF x<x8 THEN x=x8
650 :   IF x>x9 THEN x=x9
660 :   IF y<y8 THEN y=y8
670 :   IF y>y9 THEN y=y9
680 :   i% = i8 + mx * (x-x8)
690 :   j% = j8 + my * (y-y8)
700 :   MOVE i0, j0: DRAW i%, j%, 1
710 :   REM--- MOVE i0, j0:
               DRAW (i0+i%)/2, (j0+j%)/2, 1
720 :   REM--- PLOT i%, j%, 1
730 :   k = k+1
740 :   IF k MOD 10 = 0 THEN MOVE i%-6, j%:
        DRAW i%, j%+6, 1: DRAW i%+6, j%, 1:
        DRAW i%, j%-6, 1: DRAW i%-6, j%, 1
750 :   i0 = i%: j0 = j%
760 WEND
765 RAD
770 :
775 a$=INKEY$: IF a$="" THEN 775
777 CLS #0
```

```
780 LOCATE 50,3:PRINT"weiter mit   run <ENTER>"
790 LOCATE 1,1: LIST 180-200,#0
```

Die beiden Differentialgleichungen sind in dem Unterprogramm 190—210 einzugeben. Als Programmvariable für dx/dt ist xt und für dy/dt ist yt zu verwenden. Die im Unterprogramm verwendeten Werte v0 (Anfangsgeschwindigkeit) und w0 (Steigungswinkel) sind am Programmbeginn zu besetzen (Zeile 180). Weil wir die Wurfparabel von t0=0 Sekunden bis t9=5 Sekunden verfolgen wollen, geben wir diese Werte in der Zeile 230 ein. Der Wert dt entspricht der Schrittweite. Als Abschußpunkt wählen wir x0=10m,y0=10m (Zeile 240).

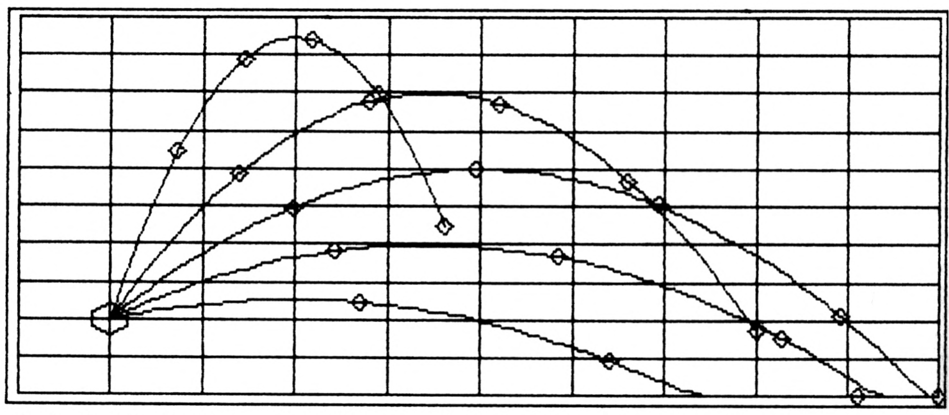
Da wir mehrere Wurfparabeln in ein Bild zeichnen wollen, arbeitet das Programm mit Fenstern. In Zeile 310 wird ein Textfenster eingerichtet, in dem die Ein- und Ausgabe von Programmtext stattfindet. Auch die Meldungen des Rechners werden in diesem Fenster ausgegeben, das mit #0 angesprochen wird. Dadurch bleibt die Grafik unversehrt. Das Fenster, in das die Grafik gezeichnet werden soll, wird durch die Zeile 410 eingeschaltet. Von nun an beziehen sich alle Grafikbefehle auf dieses Grafikfenster. Unser Bild soll die x-Werte von 0 bis 100m und die y-Werte von 0 bis 50m enthalten (Zeile 250). Nachdem die Maßstäbe mx,my und der Bildpixel (i0,j0) des Punktes P0(x0,y0) in den Zeilen 280,290 berechnet sind, wird die alte Grafik gelöscht (Zeilen 390—410) und der Abschußpunkt P0 durch ein Sechseck markiert (Zeilen 460—490). Die Zeilen 510 bis 540 zeichnen ein Gitternetz, mit dessen Hilfe die Bahnpunkte maßstäblich entnommen werden können. Durch die Zeilen 620,630 wird ein neuer Bahnpunkt ermittelt, der durch die Zeilen 680,690 in die Bildpixels umgerechnet wird. Die Zeile 700 zeichnet eine Linie vom „alten“ zum „neuen“ Kurvenpunkt.

Will man die Kurve gestrichelt zeichnen, so ist die Zeile 700 zu neutralisieren (hinter die Zeilennummer 700 ein „REM“ einzufügen) und die Zeile 710 zu aktivieren. Kurven können auch punktiert dargestellt werden, wenn die Zeilen 700,710 neutralisiert und 720 aktiviert wird. Durch die Zeile 740 wird jeder 10. Kurvenpunkt durch ein kleines Quadrat gekennzeichnet. Weil wir für dt=0.1 Sekunden gewählt haben, kennzeichnet ein solches Quadrat den Ort des Körpers im Sekundenabstand. Wir erhalten die Grafik 70, wenn wir das Programm starten (RUN) und nach der Erstellung der ersten Bahnkurve (w0 = 15°) eine Taste drücken (Zeile 775).

Nun wird der Programmanfang gelistet (Zeile 790). Wir geben in der Zeile 180 für w0 statt 15 die Zahl 30 ein. Wir starten (RUN) und zeichnen die Wurfparabel mit dem Abschußwinkel von 30° in die gleiche Grafik, bei sonst unverändeten Werten. Auf diese Weise erhalten wir für die Abschußgeschwindigkeit von v0 = 28 m/s und den Abschußwinkeln von 15°, 30°, 45°, 60°, 75° die Wurfparabeln der Grafik 70. In gleicher Weise können natürlich auch andere Werte (Abschußgeschwindigkeit, Erdbeschleunigung) variiert werden.

Natürlich spielt auch die Luftreibung eine Rolle. Ist m die Masse und v die momentane Geschwindigkeit des Wurfkörpers, so sei die Luftreibung durch die Kraft

```
F = m * c * v*v
```



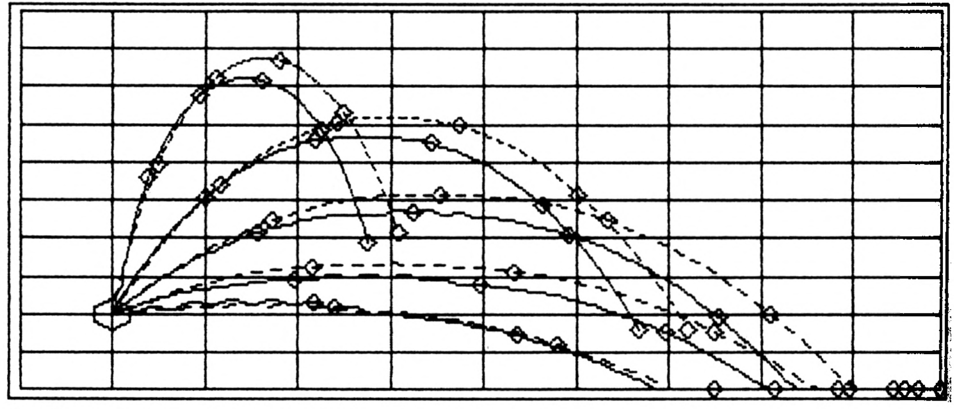
Grafik 70: Wurfparabeln mit verschiedenen Abschußwinkeln

gegeben. Diese Kraft ist stets entgegengesetzt zur Geschwindigkeit v gerichtet. Der Wert für c hängt von vielen Faktoren ab (Form des Körpers, Luftdruck, Lufttemperatur). Wir ersetzen nun die Differentialgleichungen für den reibungsfreien Wurf (Unterprogramm 190–210) durch

```
190 c = 0.1: a = c * (xt*xt + yt*yt)
195 xt = v0*COS(w0) - a*dt*SGN(xt)
200 yt = v0*SIN(w0) - a*dt*SGN(yt) - 9.81*(t-t0)
```

und erhalten die Grafik 71.

Die durchgezogenen Kurven entsprechen dem Luftwiderstandbeiwert c = 0.1 und die gestrichelten Linien dem Wert c = 0.05. Die Anfangsgeschwindigkeit ist v0 = 28m/s, und die Abschußwinkel betragen 15°, 30°, 45°, 60°, 75°.



Grafik 71: Simulation von Würfen mit unterschiedlichem Luftwiderstand

Wir wollen nun ein modernes Verfahren nach GRAGG-STOER anfügen, das die Differentialgleichungen

```
dx/dt = f(t,x,y)
dy/dt = g(t,x,y)
```

numerisch integriert. Die auf der Mittelpunktsregel beruhende GRAGG-Funktion erlaubt die Anwendung des Extrapolationsverfahrens nach STOER oder ROMBERG. Die Differenz extrapolierter Werte kann zur Fehlerabschätzung herangezogen werden. In der Zeile 260 wird die Fehlerschranke ee vorgegeben. Nach STOER-BULIRSCH: „Einführung in die Numerische Mathematik II, Springer Verlag" ist das Extrapolationsverfahren die zur Zeit effektivste Methode zur Integration einer Differentialgleichung. Die Variablenbezeichnungen sind ähnlich denen im bereits beschriebenen Programm zum Halbschrittverfahren.

```
100 REM !------------------------------------!
110 REM ! grafische Darstellung der Loesung !
120 REM !   von Differential-Gleichungen    !
130 REM !------------------------------------!
140 GOTO 230
150 REM- Unterprogramm fuer die Ableitungen -
160 REM- xt = f(t,x,y)  (xt entspricht dx/dt)-
170 REM- yt = g(t,x,y)  (yt entspricht dy/dt)-
180 :
190 xt =20 +0.2*y
200 yt =20 - 9.81*t + 0.1*x
210 RETURN
220 :
230 REM- Extrapolationsverf. nach GRAGG-STOER -
240 DIM xk(8), xx(8), yk(8), yy(8)
250 t0 = 0: t9 = 4: x0=0: y0= 10
260 nn=10: dt=(t9-t0)/nn:  ee=0.01
270 x8 =-5  : x9 =  95  : y8 =   0: y9 = 50
280 :
290 i8 = 0.5: i9 = 639.5: j8 = 0.5: j9 = 319.5
300 mx = (i9-i8)/(x9-x8): i0 = i8 + mx*(x0-x8)
310 my = (j9-j8)/(y9-y8): j0 = j8 + my*(y0-y8)
320 :
330 WINDOW #0,1,80,1,5
340 CLS #0
350 INPUT" vorbesetzte Werte aendern (j/n) ";a$
360 IF LOWER$(a$)="j" THEN LOCATE 50,3:
    PRINT"weiter mit    run 370  <ENTER>":
    LOCATE 1,1: LIST 250-270,#0
365 :
```

```
370 INPUT" alte Grafik loeschen (j/n) ";a$
380 IF LOWER$(a$)<>"j" THEN 450
385 :
390 MODE 2:BORDER 2:INK 0,1:INK 1,26
400 WINDOW #0,1,80,1,5
410 ORIGIN 0,0,i8,i9,j9,j8
420 :
430 MOVE 0,0: DRAW 639,0,1: DRAW 639,319,1
440 DRAW 0,319,1: DRAW 0,0,1
450 :
460 FOR t2=t0+dt TO t9 STEP dt: t1=t2
470 :   FOR k=1 TO 8
480 :       n=2^k: h=(t1-t0)/n: x1=x0: y1=y0
490 :
500 :       t=t0: x=x0: y=y0: GOSUB 150
510 :       x2=x0+h*xt: y2=y0+h*yt
520 :
530 :       FOR i=1 TO n-1
540 :          t=t0+i*h: x=x2: y=y2: GOSUB 190
550 :          x3=x1+2*h*xt:  y3=y1+2*h*yt
560 :          x1=x2: x2=x3:  y1=y2: y2=y3
570 :       NEXT i
580 :
590 :       t=t1: x=x2: y=y2: GOSUB 190
600 :       xk(k) = (x1+x2+h*xt)/2:
            yk(k) = (y1+y2+h*yt)/2
610 :
620 :       REM--- ROMBERG-Extrapolation ---
630 :       p=1: IF k=1 THEN 730
640 :       FOR j=k TO 2 STEP -1: p=4*p
650 :          xx(j)=(p*xk(j)-xk(j-1))/(p-1)
660 :          yy(j)=(p*yk(j)-yk(j-1))/(p-1)
670 :       NEXT j
680 :
690 :       IF k=2 THEN h1=9E+09: h3=h1
700 :       h2=xx(k): h4=yy(k)
710 :       IF ABS(h2-h1)<ee*ABS(h2) AND
            ABS(h4-h3)<ee*ABS(h4) THEN k1=k:k=99
720 :       h1=h2: h3=h4
730 :   NEXT k
740 :
```

```
750 :   IF k<99 THEN
        PRINT"Genauigkeit";ee;"nicht erreicht":
        END
760 :   REM--- PRINT t2; xx(kl); yy(kl); ee
770 :
780 :   i% = i8 + mx * (x-x8)
790 :   j% = j8 + my * (y-y8)
800 :   IF i%<0 OR i%>639 THEN 850
810 :   IF j%<0 OR j%>319 THEN 850
820 :   MOVE i0-8,j0: DRAW i0,j0+8, 1:
        DRAW i0+8,j0, 1: DRAW i0,j0-8, 1:
        DRAW i0-8,j0, 1
830 :
840 :   MOVE i0, j0: DRAW i%, j%, 1
850 :   t0=t2:x0=xx(kl):y0=yy(kl):i0=i%:j0=j%
860 NEXT t2
870 :
880 a$=INKEY$: IF a$="" THEN 880
890 CLS #0
900 LOCATE 50,3:PRINT"weiter mit   run <ENTER>"
910 LOCATE 1,1: LIST 190-200,#0
```

16.2 Differentialgleichungssysteme

Die RUNGE-KUTTA-Formeln zur numerischen Lösung von Differentialgleichungen wurden am Anfang dieses Jahrhunderts aufgestellt. GILL hat später die freien Parameter des Lösungsalgorithmus so gewählt, daß für das automatische Rechnen ein kleiner Programmieraufwand, geringer Speicherbedarf und ein effizienter Algorithmus entstand mit einem Fehler $0(h^5)$. Wir verwenden eine Variante zur Lösung von Differentialgleichungssystemen 1. Ordnung. Bei vielen Anwendungen (Prozeßrechnen, Simulationen) sind die Differentialgleichungssysteme n-ter Ordnung

```
y0' = 1
y1' = f1(x, y1, y2, ..., yn)
y2' = f2(x, y1, y2, ..., yn)
y3' = f3(x, y1, y2, ..., yn)
 .    .    .    .    .
 .    .    .    .    .
 .    .    .    .    .
yn' = fn(x, y1, y2, ..., yn)
```

zu lösen. Diese Differentialgleichungen entsprechen dem gewählten Modell. Hierbei entspricht x=y0 und y0'=1.

Auch lineare Differentialgleichungen n-ter Ordnung können auf dieses System von Differentialgleichungen zurückgeführt werden. Wir wollen dies an der linearen Differentialgleichung 4. Ordnung zeigen. Aus

```
y1'''' + p1 * y1''' + p2 * y1'' + p3 * y1' + p4 * y1 = f
```

folgt das System

```
y1' = y2
y2' = y3
y3' = y4
y4' = f - p1 * y4 - p2 * y3 - p3 * y2 - p4 * y1
```

Hierbei können p1,p2,p3,p4,f Funktionen von x=y0 sein. Dieses System von Differentialgleichungen ist ein Sonderfall des allgemeinen Differentialgleichungssystems.

Zur Lösung mit dem RUNGE-KUTTA-GILL-Algorithmus sind zunächst die Koeffizienten a0,a1,a2,a3,b0,b1,b2,b3,c0,c1,c2,c3 (Zeilen 420 bis 440) zu besetzen und die Anfangswerte y1,y2,...,yn an der Stelle x=a (Zeile 480,590 bis 610) vorzugeben. Der RUNGE-KUTTA-GILL-Algorithmus besteht dann aus den Zeilen 910 bis 970. Die Werte yi (i=1,2,...,n) der gesuchten Funktionen werden an den Stellen x=a+i*h ermittelt. Die x-Schrittweite (Zeile 500,560) ist mit h bezeichnet.

Die qi-Werte entsprechen den Fehlern bei einem Schritt. Eine automatische Schrittweitensteuerung ist nicht im Programm enthalten. Einen optischen Eindruck von den Fehlern erhält man, indem man bei gleichen Anfangswerten eine Schrittweitenänderung durchführt und diese Kurve in die gleiche Grafik zeichnet.

```
100 REM !--------------------------------!
105 REM !     Loesung eines linearen     !
110 REM !  Differentialgleichungssytems  !
115 REM !     nach  Runge-Kutta-Gill     !
120 REM !--------------------------------!
130 GOTO 230
150 REM Unterprog. fuer das DGL-System
170 ys(1) = y(3):  ys(2) = y(4)
180 ys(3) = y(2):  ys(4) = y(1)
200 RETURN
210 x8 = -0.5: x9 = 4.5: y8 = -3: y9 = 2
220 RETURN
225 :
230 WINDOW #0,1,80,1,5
240 CLS #0
250 PRINT"Eingeben des lin. DGL-Systems";:
    PRINT SPC(15)"y(0)  entspricht   x"
```

```
260 PRINT"wie z.b:   170 ys(1) = 1/y(2)";:
    PRINT SPC(15)"y(1)  entspricht  y1"
270 PRINT"               190 ys(2) =-1/y(1)";:
    PRINT SPC(15)"y(2)  entspricht  y2";
280 PRINT"  usw."
290 PRINT SPC(25)"usw.";: PRINT SPC(14)
    "ys(1)  entspricht  y1´ (1.Ableitung)"
300 PRINT"weiter:  Taste druecken!";:
    PRINT SPC(19)"ys(2)  entspricht  y2´ usw."
310 a$=INKEY$: IF a$="" THEN 310
320 :
330 CLS #0
340 LOCATE 50,5:
    PRINT"fortsetzen mit   run 400"
380 LOCATE 1,1: LIST 150-190,#0
400 DIM a(3), b(3), c(3),  y(10), ys(10),
    q(10), j1%(10), j2%(10)
410 h=1/SQR(2): ys(0)=1
420 b(0)=2:   b(1)=1:   b(2)=1:   b(3)=2
430 a(0)=1/2: a(1)=1-h: a(2)=1+h: a(3)=1/6
440 c(0)=1/2: c(1)=1-h: c(2)=1+h: c(3)=1/2
450 GOSUB 210
460 i8 = 0.5: i9 =639.5: j8 = 0.5: j9 =319.5
470 INPUT" Anzahl der eingeg. Gleichungen"; n
480 PRINT" x-Anfangswert            a=  "; x8
490 PRINT" x-Endwert                b=  "; x9
500 PRINT" x-Schrittweite   delta x=  ";
    (x9-x8)/50
510 y(0) = x8: xx = x9: h = (xx-y(0))/50
520 INPUT" Werte aendern  ";a$
530 IF LOWER$(a$)<>"j" THEN 570
540 INPUT" x-Anfangswert            a= "; y(0)
550 INPUT" x-Endwert                b= ";  xx
560 INPUT" x-Schrittweite   delta x= ";    h
570 :
580 PRINT" Eingeben der Anfangsbedingungen:"
590 FOR i=1 TO n
600 :  PRINT"   Wie gross ist  y"; i;
         "  bei  x = "; y(0);
610 :  INPUT y(i)
620 NEXT i
630 :
```

```
640 INPUT" alte Grafik loeschen (j/n) ";a$
650 CLS #0:IF LOWER$(a$)<>"j" THEN 770
660 :
680 MODE 2:BORDER 2:INK 0,1:INK 1,26
690 WINDOW #0,1,80,1,5
700 ORIGIN 0,0,i8,i9,j9,j8
710 :
720 MOVE i8,j8: DRAW i9-i8,j8, 1
725 DRAW i9-i8,j9-j8, 1: DRAW i8,j9-j8, 1
730 DRAW i8,j8, 1
740 :
750 FOR i=i8 TO i9 STEP (i9-i8)/10:
    MOVE i,j8: DRAW i,j9,1: NEXT
760 FOR j=j8 TO j9 STEP (j9-j8)/10:
    MOVE i8,j: DRAW i9,j,1: NEXT
770 :
780 mx=(i9-i8)/(x9-x8): my=(j9-j8)/(y9-y8)
790 i1% = i8 + mx*(y(0)-x8):
    IF i1%<0 OR i1%>639 THEN STOP
800 :
810 FOR i=1 TO n
820 :  j1%(i) = j8 + my*(y(i)-y8)
830 NEXT i
840 :
850 REM--- CLS #0
860 REM--- PRINT " x", " y(i)", " q(i)"
870 REM--- PRINT y(0), y(1), q(1)
880 REM--- IF n>1 THEN FOR i=2 TO n:
           PRINT  ,y(i), q(i): NEXT i
885 :
890 IF h>0 AND y(0)>=xx THEN 1160
900 IF h<0 AND y(0)<=xx THEN 1160
905 :
910 FOR j=0 TO 3: GOSUB 170
920 :  FOR i=0 TO n
930 :     h1 = a(j) * (ys(i) - b(j)*q(i))
940 :     y(i) = y(i) + h*h1
950 :     q(i) = q(i) + 3*h1 - c(j)*ys(i)
960 :  NEXT i
970 NEXT j
980 :
990 i2% = i8 + mx * (y(0)-x8)
```

```
1000 IF i2%<0 OR i2%>639 THEN 1140
1010 :
1020 FOR i=1 TO n
1030 :   j2%(i) = j8 + my * (y(i)-y8)
1040 :   IF j1%(i)<0 OR j1%(i)>319 THEN 1130
1050 :   IF j2%(i)<0 OR j2%(i)>319 THEN 1130
1060 :   IF k/5<>INT(k/5) THEN 1110
1070 :   d = 4+i
1080 :   MOVE i1%-d,j1%(i):DRAW i1%,j1%(i)+d,1
1090 :   DRAW i1%+d,j1%(i),1:DRAW i1%,j1%(i)-d,1
1100 :   DRAW i1%-d,j1%(i),1
1105 :
1110 :   MOVE i1%,j1%(i): DRAW i2%,j2%(i),1
1120 :   j1%(i) = j2%(i)
1130 NEXT i
1140 i1% = i2%: k=k+1
1150 GOTO 850
1160 a$=INKEY$: IF a$="" THEN 1160
```

Wir beschreiben nun das Programm zur Lösung von Differentialgleichungssystemen. In den Zeilen 150 bis 200 werden die Differentialgleichungen

```
y1' = f1 (y0, y1, y2, ..., yn)
y2' = f2 (y0, y1, y2, ..., yn)
 . . . . . .
 . . . . . .
 . . . . . .
yn' = fn (y0, y1, y2, ..., yn)
```

eingetragen. Hierbei wird z. B. für y2' geschrieben ys(2). Die Variable x ist im Programm als y(0) zu schreiben.

Die Zeilen 250 bis 380 dienen lediglich zur Erklärung und der Aufforderung, in dem Unterprogramm von 150 bis 200 das aktuelle Differentialgleichungssystem als Programmzeilen einzugeben. Danach wird das Programm mit RUN 400 fortgesetzt. In den Zeilen 410 bis 440 werden die für das RUNGE-KUTTA-GILL-Verfahren erforderlichen Konstanten besetzt. In der Zeile 470 wird die Anzahl n der eingegebenen Differentialgleichungen abgefragt. Dann wird der x-Anfangswert y(0),x-Endwert xx und die x-Schrittweite h eingegeben. Durch die Zeilen 580 bis 620 werden die Anfangswerte von y1,y2,...,yn für den x-Anfangswert abgefragt.

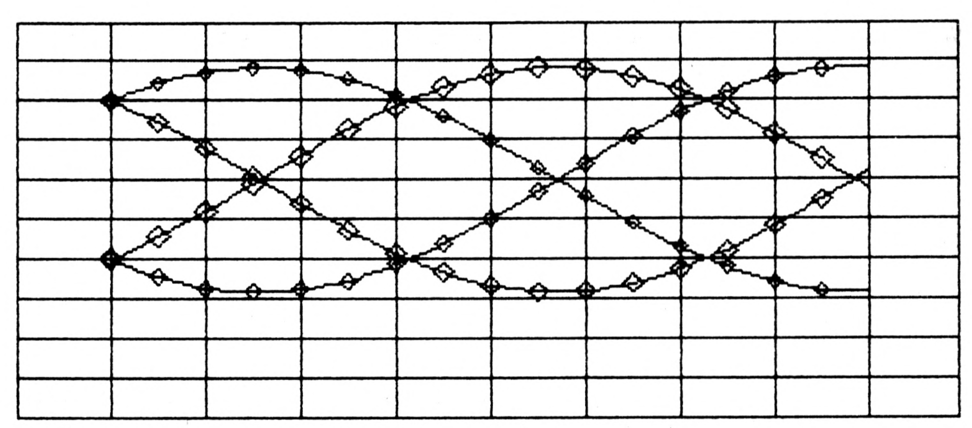
Oft möchte man mehrere Variationen in das gleiche Bild zeichnen. Deshalb wird in der Zeile 640 gefragt, ob die alte Grafik gelöscht werden soll. Beim erstmaligen Zeichnen ist das alte Bild zu löschen. Durch MODE 2 (Zeile 680) wird der hochauflösende Bildschirm aktiviert. Die Zeilen 750,760 zeichnen ein Gitternetz, das einer Einteilung des x- und y-Berei-

ches in 10 Teilintervalle entspricht. In der Zeile 780 werden die Zeichenmaßstäbe mx,my berechnet. Der zu dem x-Anfangswert gehörende Pixel i1% wird in der Zeile 790 berechnet. Die Schleife (Zeilen 810 bis 830) rechnet alle y-Anfangswerte in die Pixel j1%(i) um.

Die Zeilen 850 bis 880 können aktiviert werden, wenn eine Tabellenausgabe der berechneten Werte gewünscht wird. Die Zeilen 890,900 beenden den Programmlauf, wenn das Ende des x-Zeichenbereiches erreicht ist. Die Zeilen 910 bis 970 stellen den RUNGE-KUTTA-GILL-Algorithmus dar. Für die um h versetzte Stelle wird in der Zeile 990 der „neue" Pixel i2% und im folgenden (Zeilen 1020 bis 1130) die „neuen" y-Pixels j2%(i) ermittelt und die n-Kurven stückweise gezeichnet (Zeile 1110).

Damit man die Kurven y1(x),y2(x),. . .,yn(x) voneinander unterscheiden kann, werden diese durch kleine Quadrate gekennzeichnet (Zeilen 1080 bis 1100). Wegen des Wertes von d=4+i (Zeilen 1070,1080 bis 1100) erhält y1(x) infolge i=1 die kleinsten Quadrate. Die Anzahl der bereits durchgeführten x-Schritte wird durch k (Zeile 1140) hochgezählt. Lediglich in jedem fünften Schritt (Zeile 1060) werden die Kurven durch Quadrate gekennzeichnet.

Bei der Erstellung von Grafiken als Lösungen von Differentialgleichungssystemen ist darauf zu achten, daß am Anfang die y-Grenzen (min y8, max y9, Zeile 210) nicht zu eng gewählt werden, weil die Funktionen allzuleicht den vorgesehenen Bereich verlassen.



Grafik 72: Homogene Differentialgleichung 4. Ordnung

Die Grafik 72 ergibt sich entsprechend dem abgedruckten Programm für das Differentialsystem

```
170 ys(1) = y(3): ys(2) = y(4)
180 ys(3) = y(2): ys(4) = y(1)
200 RETURN
210 x8 = -0.5: x9 = 4.5: y8 = -3: y9 = 2
```

mit n=4 und dem x-Anfangswert 0, dem x-Endwert von 4, der x-Schrittweite von 0.05 und den Anfangswerten

```
y(1) =  1 (y1 =  1 für x=0)
y(2) = -1 (y2 = -1 für x=0)
y(3) =  1 (y3 =  1 für x=0)
y(4) = -1 (y4 = -1 für x=0)
```

Das hier zum Testen gewählte Differentialgleichungssystem entspricht der homogenen Differentialgleichung 4. Ordnung

```
y1'''' - y1 = 0,
```

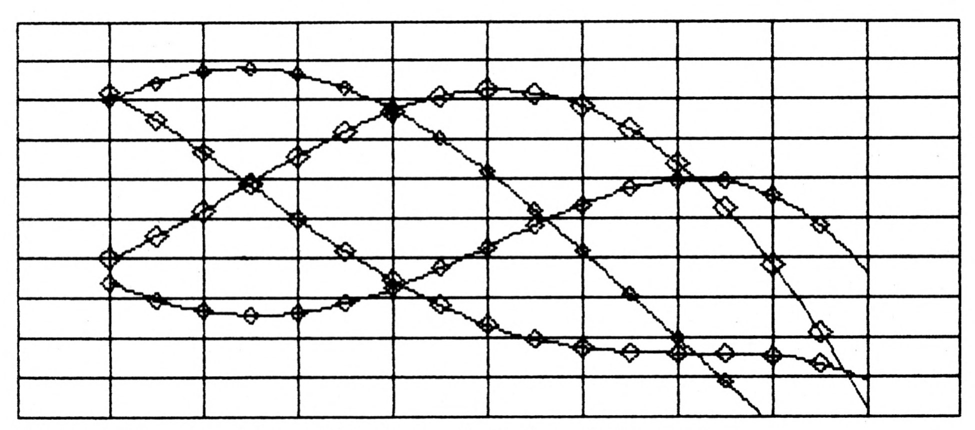
dessen allgemeine Lösung durch

```
y1(x) = k1*exp(x) + k2*exp(-x) + k3*cos(x)
+ k4*sin(x)
```

gegeben ist. Ändern wir die Anfangswerte in

```
y(1) =  1.0 (y1 =  1.0 für x = 0)
y(2) = -1.3 (y2 = -1.3 für x = 0)
y(3) =  1.1 (y3 =  1.1 für x = 0)
y(4) = -1.0 (y4 = -1.0 für x = 0)
```

ab, so ergibt sich bei sonst ungeänderten Eingaben die Grafik 73.



Grafik 73: Homogene Differentialgleichung mit geänderten Anfangswerten

16.3 Universelles Zeichenprogramm

Das im folgenden beschriebene Programm kann für unterschiedliche Anwendungen eingesetzt werden. Wir werden hier nur die wichtigsten Leistungen dieses Programmes beschreiben.

Es können Funktionen y(x) wie z. B.

```
1280 y = x*x - 4
```

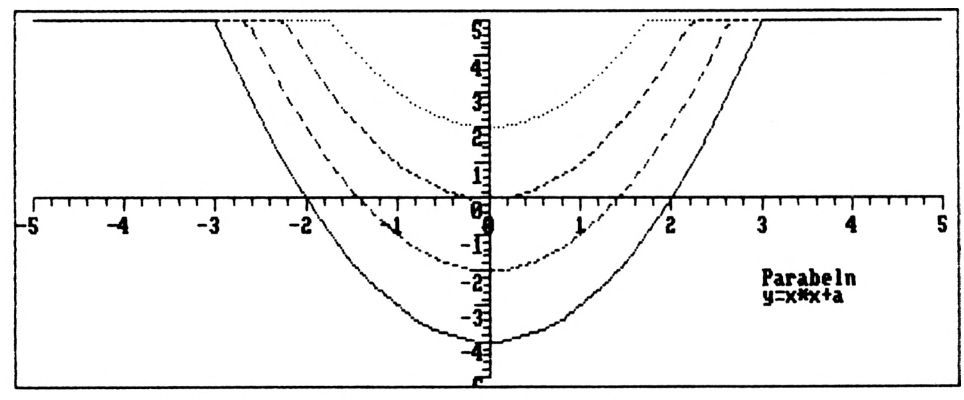
maßstäblich in ein kartesisches Koordinatensystem gezeichnet werden (Grafik 74). Die x,y-Achsen werden beschriftet. Hierbei ist das Zeichenintervall auf der x,y-Achse wählbar. Das Programm arbeitet intern mit der Parameterdarstellung von Funktionen. Deshalb sind auch Spiralen (Grafik 75) wie z. B.

```
1270 x = EXP(-0.02*t) * COS(t)
1280 y = EXP(-0.01*t) * SIN(t)
```

sowie zufallsabhängige Funktionen (Grafik 76) wie z. B.

```
1270 x = t:h = RND(1): IF h<0.5 THEN y = x+1
1280 IF h<0.5 THEN y = x-1
```

grafisch darstellbar.



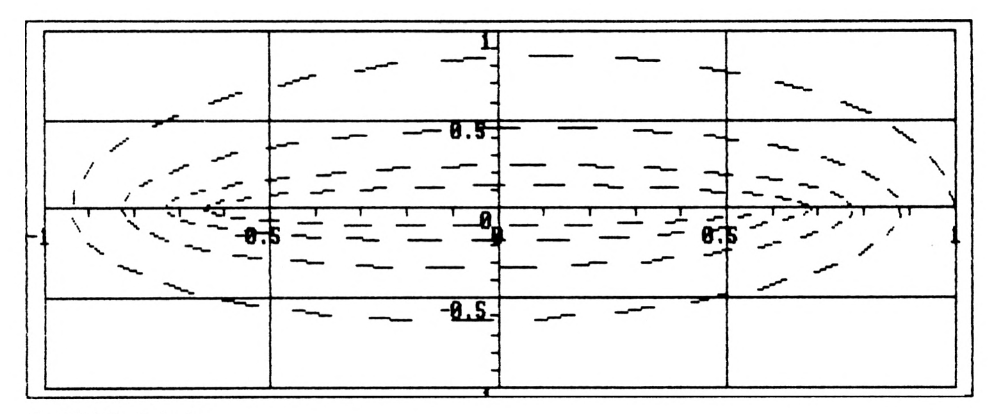
Grafik 74: Funktion y=x∗x+a *im kartesischen Koordinatensystem*

Weiterhin kann mit diesem Programm eine Punktetabelle für die Koordinaten von Meßpunkten erstellt und korrigiert werden. Diese Punkte werden mit dem Programm grafisch dargestellt, und es kann ein zugehöriges Approximationspolynom ermittelt werden. Das Polynom wird in die gleiche Grafik wie die Meßpunkte gezeichnet. Die Meßpunkte werden im allgemeinen nicht alle auf der gesuchten Approximationskurve liegen.

Ein Approximationspolynom minimiert die y-Fehlerquadratsumme.

Mit diesem Programm können auch mehrere Funktionskurven in das gleiche Bild gezeichnet werden. Durch das Eingeben der neuen Funktion in eine BASIC-Zeile (1280) und das erneute Starten des Programmes durch GOTO 2700 (nicht RUN !) bleiben die Variableninhalte zugänglich. Dadurch wird ein angenehmes interaktives Arbeiten mit Programm ermöglicht.

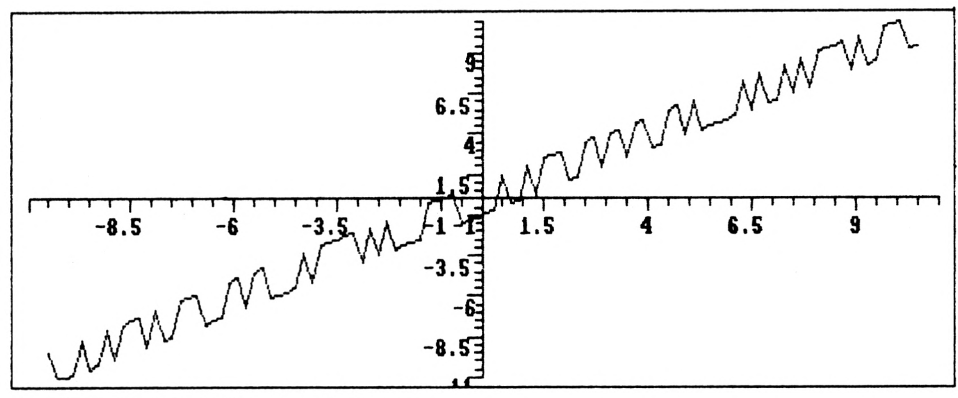
Beim Eintippen des Programmes ist **sehr sorgfältig** vorzugehen. In der Testphase ist vor jedem „RUN“ das Programm zu „SAVEn“. Insbesondere müssen auch die scheinbar unsinnigen Zeilen 1400–1530 eingegeben werden.



Grafik 75: Spiralen

Diese (und nur diese) aufgeführten Variablen werden initialisiert und sind dann im Programm verwendbar. Der Benutzer kann bei der Eingabe der zu zeichnenden Funktion die Hilfsvariablen h0,h1,h2,. . .,h8 als kurzzeitige Zwischenspeicher (1270–1280) verwenden.

Bei einem Kaltstart des Programmes ist lediglich „RUN", gefolgt von der ENTER-Taste, einzugeben. Ist jedoch nach ungewollter Beendigung des Programmes ein Warmstart erforderlich, so ist „GOTO 2400", gefolgt von ENTER, einzugeben. Das Programm verzweigt zum Hauptmenü, und die Variableninhalte sind weiterhin zugänglich.



Grafik 76: Zufallsabhängige Funktionen

Das Hauptmenü (2400–2510) ermöglicht die Auswahl von verschiedenen Leistungen. Drücken wir z. B. auf die Taste „4", so verzweigt das Programm (siehe Zeile 2550) zum Zeichnen von Funktionen (Programmzeile 2600). Der Benutzer wird aufgefordert, die zu zeichnende Funktion einzugeben. Soll die Funktion y(x) in ein x,y-System gezeichnet werden, so ist durch

```
1270 x = t
```

für x die Laufvariable t (Parameterdarstellung) zu verwenden.

In der Zeile 1280 wird nun die gewünschte Funktion wie z. B.

```
1280 y = x*x - 4
```

hinterlegt und durch „GOTO 2700“ der Programmlauf fortgesetzt.

Für die Wahl der Achsengrenzen ist die Kenntnis des kleinsten und größten Funktionswertes wünschenswert. Deshalb wird der Benutzer gefragt, ob eine solche Min-Max-Rechnung durchzuführen sei. Für diese Rechnung ist t-min, t-max und delta-t einzugeben. Für Funktionen y(x) gilt: t-min entspricht x-min, t-max entspricht x-max und delta-t entspricht delta-x.

Das Programm erfragt nun, ob der Funktionsgraph in das alte Bild gezeichnet werden soll, oder ob das alte Bild vor dem Zeichnen zu löschen ist (Zeile 3280). Dann fragt das Programm nach der Beschriftung der x- und y-Achse. Dem Benutzer werden Werte vorgeschlagen, die natürlich geändert werden können. Die Eingabewerte werden in der Zeile 3610 überprüft. Die Eingabe wird wiederholt, wenn für eine „delta“-Abfrage der Wert 0 eingegeben wird. Durch die Programmzeilen 3650,3660 wird geprüft, ob die Koordinatenachsen in die Bildmitte gezeichnet werden können. Die Zeile 3640 legt die Bildgröße in Bildschirmpixel fest. In der Zeile 3670 werden die Maßstäbe x4,y4 für die x,y-Achsen berechnet. Wird später eine zusätzliche Kurve in das gleiche Bild gezeichnet, so werden erneut diese alten Werte x4,y4,i8,j8,x8,y8 zur Umrechnung in die Bildschirmpixel verwendet (siehe Zeilen 1350,1360). Durch die Zeile 3770 wird die x-Achse ohne Markierung gezeichnet. Die Markierung und die Beschriftungen der x-Achse erfolgen durch die Schleife 3780 bis 3870. In ähnlicher Weise erstellen die Zeilen 3890—3990 die y-Achse. Durch die Zeilen 4050—4140 wird der Funktionsgraph gezeichnet.

Gewöhnlich ist pp=1 (Zeile 4090). Die Kurve wird durch Sehnenstücke gebildet. Für pp=0 wird die Kurve gelöscht. Für pp=3 besteht die Kurve aus Einzelpunkten. Mit Hilfe von pp können verschiedene Zeichen-Strich-Arten gewählt werden. Der Aufruf GOSUB 1270 in der Zeile 4070 bewirkt die Berechnung der x,y-Werte der eingegebenen Funktion und GOSUB 1310 die Umrechnung von x,y in die Bildschirmpixel x1,y1. Außerdem prüft GOSUB 1310, ob die x,y-Werte innerhalb des (x8,y8),(x9,y9)-Rechteckes liegen. Wandert die Kurve über den Bildrand, so werden die herausfallenden x,y-Werte durch die entsprechenden Bildrandwerte ersetzt.

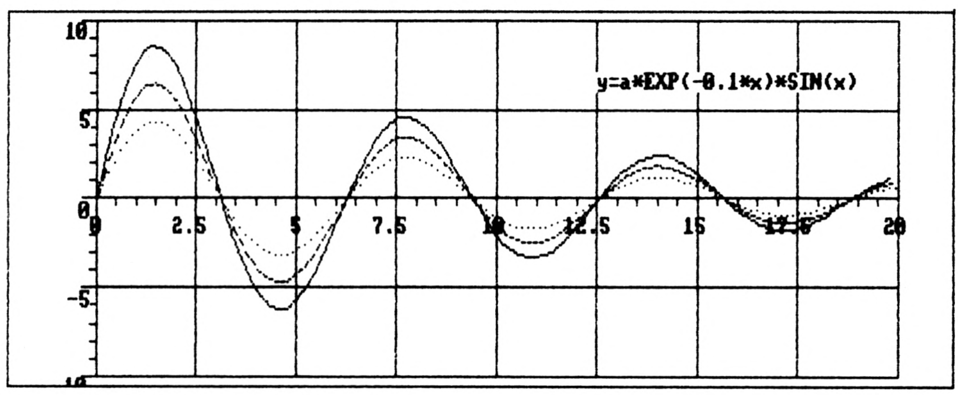
Als Beispiel wollen wir nun die Eingaben für die Grafik einer gedämpften Schwingung zeichnen lassen (Grafik 77). Nach dem Start des Programmes wählen wir im Hauptmenü die Nummer „4“ und geben

```
1270 x = t
1280 y = 10*EXP(-0.1*x) * SIN(x)
```

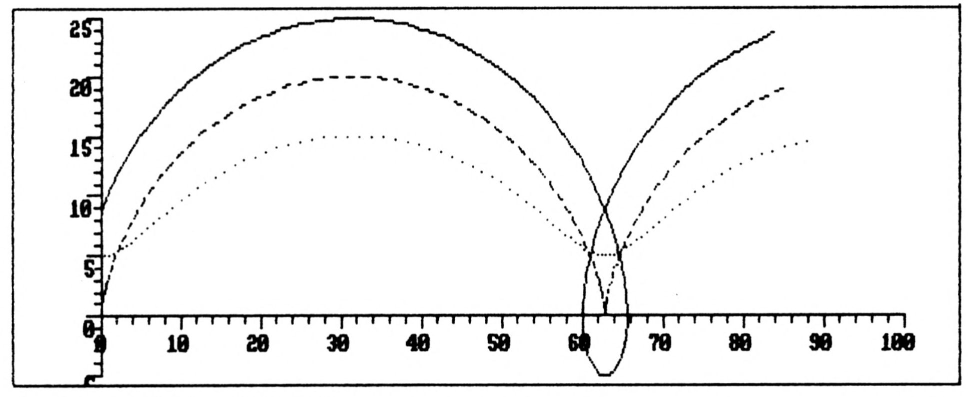
ein. Durch GOTO 2700 wird das Programm fortgesetzt. Die Min-Max-Berechnung fragt nach t-min. Weil t-min hier dem kleinsten x-Wert entspricht, geben wir 0 ein; für t-max die Zahl 20 und 0.2 für die Schrittweite delta-t der Funktionswertberechnungen. Mit der Schrittweite von 0.2 werden alle Funktionswerte durchgerechnet. Die kleinsten und größten x,y-Werte werden angezeigt. Wir löschen die alte Grafik. Für die x-Achsenbeschriftungen

geben wir 0 für x-min, 20 für x-max und 0.5 für delta-x ein. Die x-Achsenmarkierungen erfolgen also mit einer Schrittweite von 0.5.

Da die Extremwerte y-min=−6.25,y-max=8.57 ermittelt wurden, können die Achsenabschnitte geeignet gewählt werden. Für die y-Achsen-Beschriftung geben wir −10 für y-min,+10 für y-max und 1 für delta-y ein. Wir wählen eine Darstellung mit Gitternetz. Nachdem die Kurve gezeichnet ist, erscheint wieder das Hauptmenü.



Grafik 77: Gedämpfte Schwingung



Grafik 78: Zykloide und Trochoide

Weil wir eine weitere gedämpfte Schwingung in die gleiche Grafik zeichnen wollen, wählen wir die „4“. Nun ändern wir die Zeile 1280 in

```
1280 y = 7.5*EXP(-0.1*x) * SIN(x)
```

ab. Nach dem Fortsetzen mit GOTO 2700 braucht die Min-Max-Berechnung nicht erneut ausgeführt werden. Die t-Grenze legen wir durch 0,20,0.15 fest. Da wir die neue Kurve in die alte Grafik zeichnen wollen, darf das alte Bild nicht gelöscht werden. Nachdem die Kurve gezeichnet ist, wird in gleicher Weise die 3. Funktion

```
1280 y = 5*EXP(-0.1*x) * SIN(x)
```

in das gleiche Koordinatensystem gezeichnet. Wir erhalten die Grafik 77. Wie zu erkennen ist, wurde für jede Kurve jeweils eine andere Linienart gewählt. Die Linienart 2 ergibt bei unterschiedlicher Schrittweite delta-t verschiedene Strichlängen. Für die Linienart 3 (punktiert) wird der PLOT-Befehl (Zeile 4120) verwendet.

In ähnlicher Weise können wir z. B. die Zykloide und Trochoide

```
1270 b = 15: a = 10
1275 x = a*t - b*SIN(t)
1280 y = a - b*COS(t)
```

zeichnen lassen. Für die punktierte Kurve der Grafik 78 gilt a=10,b=5, für die gestrichelte Linie gilt a=10,b=10, und für die durchgezogene Linie gilt a=10,b=15.

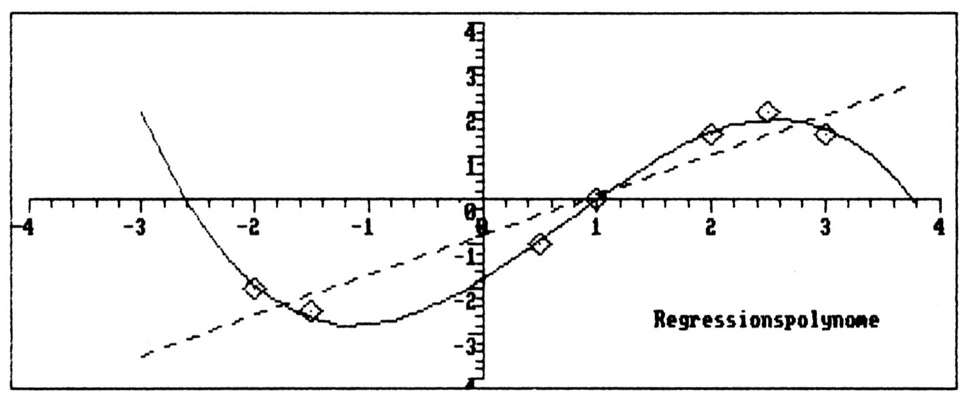
Mit diesem Zeichenprogramm können auch Approximationspolynome berechnet und gezeichnet werden. Hierzu starten wir das Programm und wählen beim Hauptmenü die Zahl 1 (Punktetabelle eingeben). Wir beginnen mit der Punktnummer 1 (1 gefolgt von ENTER eingeben) und geben für x den Wert −2 gefolgt von ENTER und für y den Wert −2 ein. Auf diese Weise geben wir die folgenden Koordinaten

Nr.	1	2	3	4	5	6	7
x	−2.0	−1.5	+0.5	+1.0	+2.0	+2.5	+3.0
y	−2.0	−2.5	−1.0	0.0	+1.5	+2.0	+1.5

ein. Liegt ein Eingabefehler vor, so geben wir am Zeilenanfang die entsprechende Punktnummer, gefolgt von den koorigierten x,y-Werten, ein. Eine negative Punktnummer, gefolgt von ENTER, beendet die Eingabe.

Wählen wir beim Hauptmenü die Zahl 2, so wird die vollständige Tabelle angezeigt. Wird die Zahl 5 gewählt, so verzweigt das Programm zum Zeichnen dieser Meßpunkte. Die Min-Max-Berechnung ergibt x-min=−2,x-max=3,y-min=−2.5,y-max=2 und fragt nach der Kennzeichnung der Punkte. Wir wählen die Punktekennung 1 (Quadrate) und löschen das alte Bild. Für die x-Achsenbeschriftung geben wir −4 für x-min,+4 für x-max und 0.2 für delta-x ein sowie für y-min −4, für y-max +4 und für delta-y 0.2. Wir wählen eine Darstellung ohne Gitternetz (Grafik 79). Nun wird das Koordinatensystem gezeichnet, und die Punkte aus der Punktetabelle werden eingetragen.

Wir wollen abschließend ein Approximationspolynom berechnen lassen, das die 7 Meßpunkte möglichst gut beschreibt. Hierzu wählen wir am Hauptmenü die Zahl 8 (Regressionspolynom). Als Polynomgrad wählen wir 1. Dieses Polynom entspricht der Regressionsgeraden. Der Rechner zeigt an, daß diese Geradengleichung in der Zeile 1280 einzugeben ist. Wir übernehmen die Zeile 1280 mit dem COPY-CURSOR und betätigen die ENTER-Taste. Dann setzen wir durch die Eingabe von GOTO 2700, gefolgt von ENTER, den Programmlauf fort. Da wir das Koordinatensystem bereits gezeichnet haben, ist eine erneu-



Grafik 79: Approximationspolynom 1. und 3. Grades

te Min-Max-Berechnung nicht erforderlich. Der kleinste t-Wert entspricht x-min. Die Regressionsgerade soll von x=−3.0 bis x=4.0 gezeichnet werden. Deshalb geben wir für t-min den Wert −3, für t-max den Wert 4 und für delta-t den Wert 0.2 ein. Die Regressionsgerade soll als gestrichelte Linie (2) in das alte Bild gezeichnet werden.

Wir wiederholen das Verfahren für ein Regressionspolynom 3. Grades, das wir durchgezogen in das gleiche Bild zeichnen und schreiben den Text „Regressionspolynome" ab der Stelle 1.5,−2.5 in die Grafik. Natürlich können jetzt Punkte geändert oder hinzugefügt und das dazugehörige Approximationspolynom gezeichnet werden. In den Zeilen 5500 bis 5710 ist ein Programmstück enthalten, das die Ausgabe der hochauflösenden Grafik auf einem SCHNEIDER-Drucker ermöglicht. Die Ausgabe eines Bildes in hochauflösender Grafik benötigt ca. 3 Minuten.

```
1100 REM !---------------------------!
1110 REM ! Universelles Plot-Programm !
1120 REM ! fuer die Darstellung von   !
1130 REM ! Funktionen in Parameterdar-!
1140 REM ! stellung und zum Zeichnen  !
1150 REM ! von Messpunkten sowie der  !
1160 REM ! Regressionspolynome        !
1170 REM !---------------------------!
1180 GOSUB 5500: REM zur Druckerroutine
1190 :
1200 MODE 2:BORDER 2:INK 0,1:INK 1,26
1210 WINDOW #0,1,80,1,6
1220 ORIGIN 0,0,0,639,303,0
1230 GOTO 1390
1240 :
1250 :
```

```
1260 REM- Funktion in Parameterdarstellung -
1270 x = t
1280 y = x*x - 4
1290 RETURN
1300 :
1310 IF x<x8 THEN x=x8
1320 IF x>x9 THEN x=x9
1330 IF y<y8 THEN y=y8
1340 IF y>y9 THEN y=y9
1350 x1 = i8 + x4*(x-x8)
1360 y1 = j8 + y4*(y-y8)
1370 RETURN
1380 :
1390 nn=0:pp=1:ll=1:kk=1
1400 i8=0:i9=0:i1=0:i2=0:j8=0:j9=0
1410 a=0:b=0:c=0:d=0:e=0:f=0:g=0:h=0:i=0:j=0
1420 k=0:l=0:m=0:n=0:o=0:p=0:q=0:r=0:s=0:t=0
1430 u=0:v=0:w=0:x=0:y=0:z=0
1440 t8=-10:t9=10:t5=0.2
1450 x0=0:x1=0:x2=0:x3=0:x4=0:x5=0:x6=0:x7=0
1460 x8=0:x9=0
1470 y0=0:y1=0:y2=0:y3=0:y4=0:y5=0:y6=0:y7=0
1480 y8=0:y9=0
1490 h0=0:h1=0:h2=0:h3=0:h4=0:h5=0:h6=0:h7=0
1500 h8=0:h9=0
1510 n0=1:n1=1:n2=1:n3=1:n4=100:n5=1:n6=1:n7=1
1520 n8=1:n9=n4
1530 a$="":b$="":h$="":jn$=""
1540 DIM i(8), x(100), y(100), a(102,7)
1550 FOR i=1 TO n4: x(i)=-i: y(i)=-i: NEXT
1560 GOTO 2400
1570 :
1600 REM--- Punkte Eingeben ---
1610 CLS#0
1620 i=0
1630 IF n4>n9-1 THEN n4=1
1640 PRINT#0,
     "          Nr negativ beendet die Eingabe !"
1650 PRINT#0,
     "    Nr            X             Y"
1660 i=i+1: IF i<1 OR i>n9 THEN 2400
1670 INPUT#0,;"  ";i:IF i<1 OR i>50 THEN 1730
```

```
1680 INPUT#0,;"              ."; x(i)
1690 INPUT#0,;"              "; y(i): PRINT#0
1700 IF ABS(x(i))+ABS(y(i)) > 9E+09 THEN 1730
1710 IF i>n4 THEN n4=i
1720 GOTO 1660
1730 GOTO 2400
1740 :
1750 REM--- Punkte Ausgeben ---
1760 CLS#0
1770 PRINT#0,
     " ENTER-Taste beendet die Ausgabe,";
1780 PRINT#0,
     " LEER-Taste setzt die Ausgabe fort"
1790 PRINT#0,
     "    Nr           X           Y"
1800 FOR i=1 TO n4
1810 PRINT#0,"    ";i;"     |    ";x(i);
     "   |    ";y(i)
1820 IF i/4<>INT(i/4) THEN 1850
1830 a$=INKEY$: IF a$="" THEN 1830
1840 IF a$=CHR$(13) THEN GOTO 1870
1850 NEXT i
1860 a$=INKEY$: IF a$="" THEN 1860
1870 GOTO 2400
1880 :
1890 REM--- Tabelle Korrigieren ---
1900 CLS#0
1910 PRINT#0,
     "   Punkte einfuegen          = 1";
1920 PRINT#0,
     "    Punkte ueberschreiben    = 4"
1930 PRINT#0,
     "   Punkte loeschen           = 2";
1940 PRINT#0,
     "    beenden der Korrektur    = 5"
1950 PRINT#0,
     "   Teil-Tabelle auswaehlen = 3"
1960 jn$=INKEY$: IF jn$="" THEN 1960
1970 h9=VAL(jn$)
1980 ON h9 GOTO 2010,2130,2230,2400,2400
1990 GOTO 1890
2000 :
2010 CLS#0:PRINT#0
```

```
2020 PRINT#0,"   Die Anzahl der ";
2030 INPUT#0,"einzufuegenden Punkte ist  ";h1
2040 PRINT#0,"   Punkte sollen ";
2050 INPUT#0,"eingefuegt werden ab der Nr. ";h2
2060 FOR j=1 TO h1
2070    FOR i=n9-1 TO ABS(h2) STEP -1
2080     x(i+1)=x(i): y(i+1)=y(i)
2090    NEXT i: x(h2)=0: y(h2)=0
2100 NEXT j: i=h2-1: n4=n4+h1
2110 GOTO 1630
2120 :
2130 CLS#0:PRINT#0
2140 PRINT#0," Alle Punkte von der  Nr1  ";
2150 PRINT#0,"bis zur  Nr2  sind zu loeschen";
2160 INPUT#0,"  Nr1, Nr2 ";h1,h2
2170 h1=ABS(h1): IF h2<h1 THEN 2130
2180 FOR i=1 TO n9-h2
2190  x(h1+i-1)=x(h2+i): y(h1+i-1)=y(h2+i)
2200 NEXT i
2210 n4=n4+h1-h2-1: GOTO 2400
2220 :
2230 CLS#0:PRINT#0
2240 PRINT#0,"  Alle Punkte von der  Nr1  ";
2250 PRINT#0,"zur  Nr2  sind in die aktuelle";
2260 PRINT#0," Tabelle fuer das"
2270 PRINT#0,"  Regressionspolynom zu ";
2280 PRINT#0,"uebernehmen";
2290 INPUT#0,"    Nr1, Nr2  "; n3,n4
2300 GOTO 2400
2310 :
2400 CLS #0
2410 LOCATE#0,31,1:PRINT#0,"CPC464 - GRAPHIK"
2420 PRINT#0
2430 PRINT#0,"Punkte-Tabelle eingeben    = 1";
2440 PRINT#0,"  Funktionen zeichnen = 4";
2450 PRINT#0,"   Graphik ausplotten = 7";
2460 PRINT#0,"Punkte-Tabelle anzeigen    = 2";
2470 PRINT#0,"  Punkte in Graphik   = 5";
2480 PRINT#0,"   Regressionspolynom = 8";
2490 PRINT#0,"Punkte-Tabelle korrigieren = 3";
2500 PRINT#0,"  Graphik beschriften = 6";
2510 PRINT#0,"   Beenden            = 9";
```

```
2520 jn$=INKEY$: IF jn$="" THEN 2520
2530 h9=VAL(jn$)
2540 :
2550 ON h9 GOTO 1600,1750,1890,2600,2700,
      4400,5310,4620,5720
2560 GOTO 2400
2570 :
2580 :
2600 CLS#0
2610 PRINT#0,"    Eingeben der Funktion";
2620 PRINT#0," mit den Zeilen-Nummern.";
2630 PRINT#0,"Dann fortsetzen mit der";
2640 PRINT#0,"    ENTER-Taste !!  ";
2650 PRINT#0,"Damit die Zeile";
2660 PRINT#0," uebernommen wird. ";
2670 PRINT#0,"Weiter mit   goto 2700 !! ";
2680 LIST 1270-1289
2690 :
2700 :
2710 CLS #0
2720 PRINT#0,"                 ";
2730 INPUT#0,"MIN-MAX-Berechnung  (j/n) ";jn$
2740 CLS#0
2750 IF h9=5 THEN t8=n3: t9=n4: t5=1:GOTO 2920
2760 PRINT#0,
     "  Grenzen fuer Laufvariable t": PRINT#0
2770 PRINT#0,
     "    Kleinster t-Wert        t-MIN   ";t8
2780 PRINT#0,
     "    Groesster t-Wert        t-MAX   ";t9
2790 IF LOWER$(jn$)="j" THEN t5=(t9-t8)/50
2800 PRINT#0,
     "    t-Schritt-Weite       Delta-t   ";t5
2810 INPUT#0,"  Werte aendern   (j/n) ";a$
2820 IF LOWER$(a$)<>"j" THEN 2890
2830 CLS#0
2840 PRINT#0,"  Grenzen fuer Laufvariable";
2850 PRINT#0," t eingeben":PRINT#0
2860 INPUT#0,
     "    Kleinster t-Wert        t-MIN   ";t8
2870 INPUT#0,
     "    Groesster t-Wert        t-MAX   ";t9
```

```
2880 INPUT#0,
     "    t-Schritt-Weite      Delta-t   ";t5
2890 IF (t9-t8)*t5<=0 THEN PRINT#0,
     "         F E H L E R !!!" ELSE GOTO 2920
2900 FOR z=0 TO 500:NEXT: GOTO 2740
2910 :
2920 IF jn$="n" THEN 3120
2930 :
2940 CLS#0
2950 PRINT#0,
     "    Die MIN-MAX-Berechnung ergibt:"
2960 x6=1E+30: x7=-x6: y6=x6: y7=-x6
2970 FOR t=t8 TO t9 STEP t5
2980 IF h9=5 THEN x=x(t): y=y(t): GOTO 3000
2990 GOSUB 1270
3000 IF x>x7 THEN x7=x
3010 IF x<x6 THEN x6=x
3020 IF y>y7 THEN y7=y
3030 IF y<y6 THEN y6=y
3040 NEXT t
3050 PRINT#0,"           X-MIN = ";x6;
     "          X-MAX = ";x7
3060 PRINT#0,"           Y-MIN = ";y6;
     "          Y-MAX = ";y7
3070 PRINT#0
3080 LOCATE#0,35,6
3090 PRINT#0,"weiter mit  LEER-Taste"
3100 t$=INKEY$: IF t$="" THEN 3100
3110 :
3120 CLS#0
3130 PRINT#0,"   Linienart eingeben       ";
3140 PRINT#0,"oder         Punktekennung"
3150 PRINT#0
3160 PRINT#0,"    1 = durchgezogene Linie ";
3170 PRINT#0,"            1 = Quadrate"
3180 PRINT#0,"    2 = gestrichelte Linie";
3190 PRINT#0,"              2 = Dreiecke"
3200 PRINT#0,"    3 = punktierte Linie";
3210 PRINT#0,"                3 = Kreise"
3220 PRINT#0,"    0 = Linie loeschen      ";
3230 PRINT#0,"            -1 = zum Hauptmenue";
3240 INPUT#0,"            "; pp
```

```
3250 IF pp<0 THEN 2400
3260 :
3270 CLS#0:PRINT#0:PRINT#0,"        ";
3280 INPUT#0,"Bild loeschen     (j/n)    ";jn$
3290 IF LOWER$(jn$)<>"j" AND LOWER$(jn$)<>"n"
     THEN 2400
3300 IF LOWER$(jn$)="n" THEN 4030
3310 :
3320 CLS#0
3330 PRINT#0,"  X-Achsen-Beschriftung"
3340 PRINT#0,"      X-Achsen-Beschriftung";
     "   X-MIN     ";INT(x6-1)
3350 PRINT#0,"      X-Achsen-Beschriftung";
     "   X-MAX     ";INT(x7+2)
3360 x8=INT(x6-1):x9=INT(x7+2):x5=(x9-x8)/25
3370 x5=INT(5*x5+0.5)/5: IF x5=0 THEN x5=0.2
3380 PRINT#0,"      X-Schritt-Weite         ";
     "   Delta-X  "; x5
3390 INPUT#0,"  Werte aendern  (j/n) ";a$
3400 IF LOWER$(a$)<>"j" THEN 3470
3410 CLS#0
3420 PRINT#0,"  X-Achsen-Beschriftung ";
     "eingeben !"
3430 PRINT#0,"      X-Achsen-Beschriftung";:
     INPUT#0,"     X-MIN    ";x8:x6=x8+1
3440 PRINT#0,"      X-Achsen-Beschriftung";:
     INPUT#0,"     X-MAX    ";x9:x7=x9-2
3450 PRINT#0,"      X-Schritt-Weite         ";:
     INPUT#0,"  Delta-X    "; x5
3460 :
3470 CLS#0
3480 PRINT#0,"  Y-Achsen-Beschriftung"
3490 PRINT#0,"      Y-Achsen-Beschriftung";
     "   Y-MIN     ";INT(y6-1)
3500 PRINT#0,"      Y-Achsen-Beschriftung";
     "   Y-MAX     ";INT(y7+2)
3510 y8=INT(y6-1):y9=INT(y7+2):y5=(y9-y8)/20
3520 y5=INT(5*y5+0.5)/5: IF y5=0 THEN y5=0.2
3530 PRINT#0,"      Y-Schritt-Weite        ";
     "  Delta-Y     "; y5
3540 INPUT#0,"  Werte aendern  (j/n)";a$
3550 IF LOWER$(a$)<>"j" THEN 3610
```

```
3560 CLS#0
3570 PRINT#0,"  Y-Achsen-Beschriftung ";
     "eingeben !"
3580 PRINT#0,"      Y-Achsen-Beschriftung";:
     INPUT#0,"    Y-MIN     ";y8:y6=y8+1
3590 PRINT#0,"      Y-Achsen-Beschriftung";:
     INPUT#0,"    Y-MAX     ";y9:y7=y9+2
3600 PRINT#0,"      Y-Schritt-Weite      ";:
     INPUT#0," Delta-Y     "; y5
3610 IF (x9-x8)*x5<=0 OR (y9-y8)*y5<=0 THEN
     PRINT#0,"  F E H L E R !!!" ELSE GOTO 3630
3620 FOR z=0 TO 500:NEXT: GOTO 2700
3630 :
3640 i8=60.5: i9=600.5: j8=38.5: j9=294.5
3650 x0=x8: IF x8<0 AND x9>0
     THEN x0=0: i8=12.5: i9=626.5
3660 y0=y8: IF y8<0 AND y9>0
     THEN y0=0: j8=8.5
3670 x4=(i9-i8)/(x9-x8): y4=(j9-j8)/(y9-y8)
3680 x=x0: y=y0: GOSUB 1350: x0=x1: y0=y1
3690 INPUT#0,"    Gitternetz  (j/n)  ";a$
3700 CLS#0
3710 IF a$<>"j" AND a$<>"n" THEN 2400
3720 IF LOWER$(jn$)="n" THEN 3740
3730 CLG 0:MOVE 0,0:DRAW 639,0,1:
     DRAW 639,303,1:DRAW 0,303,1:DRAW 0,0,1
3740 :
3750 TAG
3760 :
3770 MOVE i8,y0: DRAW i9,y0,1: j=-1
3780 FOR x=x8 TO x9 STEP x5
3790 :   x=ROUND(x,5)
3800 :   h$ = STR$(x)
3810 :   GOSUB 1350: h=6*LEN(h$): j=j+1
3820 :   MOVE x1,y0: DRAW x1,y0-5,1
3830 :   IF j/5<>INT(j/5) THEN 3870
3840 :   MOVE x1,y0: DRAW x1,y0-10,1
3850 :   IF h<50 AND h<x1
         THEN MOVE x1-h,y0-15: PRINT h$;
3860 :   IF a$="j" THEN MOVE x1,j8:DRAW x1,j9,1
3870 NEXT x
3880 :
```

```
3890 MOVE x0,j8: DRAW x0,j9,1: j=-1
3900 FOR y=y8 TO y9 STEP y5
3910 :   y=ROUND(y,5)
3920 :   h$= STR$(y)
3930 :   GOSUB 1350: h=10*LEN(h$): j=j+1
3940 :   MOVE x0,y1: DRAW x0-5,y1,1
3950 :   IF j/5<>INT(j/5) THEN 3990
3960 :   MOVE x0,y1: DRAW x0-10,y1,1
3970 :   IF h<50 AND h<x0
         THEN MOVE x0-h,y1-3: PRINT h$;
3980 :   IF a$="j" THEN MOVE i8,y1:DRAW i9,y1,1
3990 NEXT y
4000 :
4010 TAGOFF
4020 :
4030 :   IF h9 = 5 THEN 4200
4040 :
4050 REM--- Funktion zeichnen ---
4060 FOR t=t8 TO t9 STEP t5
4070 :   GOSUB 1270: GOSUB 1310
4080 :   IF t=t8 THEN 4130
4090 :   IF pp=1 THEN
         MOVE x2,y2: DRAW x1,y1,1: GOTO 4130
4100 :   IF pp=2 THEN MOVE x2,y2:
         DRAW (x2+x1)/2,(y2+y1)/2,1: GOTO 4130
4110 :   IF pp=0 THEN
         MOVE x2,y2: DRAW x1,y1,0: GOTO 4130
4120 :   PLOT x1,y1,1
4130 :   x2 = x1: y2 = y1
4140 NEXT t: GOTO 4380
4150 :
4200 REM--- Punkte zeichnen ---
4210 FOR t=t8 TO t9
4220 :   x=x(t): y=y(t): GOSUB 1310
4230 :   IF pp<>1 THEN 4270
4240 :   MOVE x1-8,y1: DRAW x1,y1+8,1
4250 :   DRAW x1+8,y1,1: DRAW x1,y1-8,1
4260 :   DRAW x1-8,y1,1
4270 :   IF pp<>2 THEN 4300
4280 :   MOVE x1-8,y1-8: DRAW x1,y1+8,1
4290 :   DRAW x1+8,y1-8,1: DRAW x1-8,y1-8,1
4300 :   IF pp<>3 THEN 4350
```

```
4310 :   DEG: FOR a=0 TO 360 STEP 10
4320 :   a2 = x1+10*COS(a): b2 = y1+10*SIN(a)
4330 :    IF a>0 THEN MOVE a1,b1: DRAW a2,b2,1
4340 :   a1 = a2: b1 = b2: NEXT a: RAD
4350 :   PLOT x1,y1,1
4360 NEXT t
4370 :
4380 GOTO 2400
4390 :
4400 REM--- Beschriften ---
4410 CLS#0
4420 PRINT#0,"     X MIN = "; x8;
     "        Y MIN = "; y8
4430 PRINT#0,"     X MAX = "; x9;
     "        Y MAX = "; y9
4440 INPUT#0,"  Ausgabetext   ";h$
4450 PRINT#0,"                        ";
4460 INPUT#0," bei   X,Y =  "; x3, y3
4470 x=x3: y=y3: GOSUB 1350
4480 :
4490 PRINT#0,"                  ";
4500 INPUT#0,"schreiben/loeschen  (s/l)  ";jn$
4510 TAG
4520 IF LOWER$(jn$)="s"
     THEN MOVE x1,y1: PRINT h$;
4530 IF LOWER$(jn$)<>"l" THEN GOTO 4580
4540 d = LEN(h$): h$=""
4550 FOR e=1 TO d: h$ = " "+h$: NEXT
4560 MOVE x1,y1: PRINT h$;
4570 :
4580 TAGOFF
4590 GOTO 2400
4600 :
4610 :
4620 REM--- Regressionspolynom ---
4630 CLS#0:PRINT#0,"      ";
4640 INPUT#0,"Grad des Polynoms         ";n2
4650 n2=n2+1: IF n2>n4-n3+1 THEN 4620
4660 PRINT#0:PRINT#0,"   Loesung des ";
4670 PRINT#0,"Gleichungssystems ..."
4680 GOSUB 5020
4690 PRINT#0,"   X          Y       ";
```

```
4700 PRINT#0,"    Fehler"
4710 FOR t=1 TO n1
4720 :  x = a(t,2): GOSUB 5230
4730 :  PRINT#0,"  ";x, a(t,0), a(t,0)-y
4740 :  IF t/5<>INT(t/5) THEN 4780
4750 :  LOCATE#0,50,6
4760 :  PRINT#0,"weiter mit LEER-Taste"
4770 :  a$=INKEY$: IF a$="" THEN 4770
4780 NEXT t
4790 PRINT#0," Polynom-Koeffizienten:"
4800 FOR i2=1 TO n2
4810 :  PRINT#0,"a(";i2-1;")="; a(0,i2)
4820 NEXT i2
4830 LOCATE#0,50,6
4840 PRINT#0,"weiter mit LEER-Taste"
4850 a$=INKEY$:IF a$="" THEN 4850
4860 PRINT#0,"   Zeilen mit COPY-CURSOR ";
4870 PRINT#0,"uebernehmen und ";
4880 PRINT#0,"ENTER-Taste betaetigen"
4890 PRINT#0,"1270 x = t"
4900 PRINT#0,"1280 y = ";
4910 FOR i2=1 TO n2: h=a(0,i2)
4920 :  IF ABS(h)<0.00000001 THEN 4960
4930 :  IF h>=0 THEN PRINT#0,"+";
4940 :  PRINT#0,h;" ";: IF i2=1 THEN 4960
4950 :  FOR i1=2 TO i2: PRINT#0,"*x";: NEXT i1
4960 NEXT i2
4970 PRINT#0
4980 PRINT#0,"                    ";
4990 PRINT#0,"           weiter mit  GOTO 2700"
5000 END
5010 :
5020 REM--- Loesung des ueberbestimmten ---
5030 REM--- Gleichungs-Systems a(i1,i2) ---
5040 REM--- Loesungen stehen in a(0,i2) ---
5050 REM--- i1=1,2,..,n1: i2=1,2,..,n2  ---
5060 n1=n4-n3+1
5070 FOR i1=1 TO n1: a(i1,0)=y(i1):
     a(i1,1)=1: a(i1,2)=x(i1)
5080 FOR i2=3 TO n2:
     a(i1,i2) = a(i1,2)*a(i1,i2-1):NEXT:NEXT
5090 FOR i1=1 TO n1: h1=0: FOR i2=1 TO n2:
     h1=h1+a(i1,i2)*a(0,i2):NEXT
```

```
5100 a(i1,n2+1)=h1-a(i1,0): NEXT: h4=1E+30
5110 FOR h5=1 TO n1+n2: h3=h4: h4=0:
     FOR i2=1 TO n2: h1=0: FOR i1=1 TO n1
5120 h1=h1+a(i1,i2)*a(i1,n2+1): NEXT:
     a(n1+2,i2)=h1: h4=h4+h1*h1: NEXT
5130 IF h3<1E-10 THEN 5220
5140 h1=h4/h3: FOR i2=1 TO n2:
     a(n1+1,i2) = -a(n1+2,i2)+h1*a(n1+1,i2)
5150 NEXT: h1=0: FOR i1=1 TO n1: h2=0:
     FOR i2=1 TO n2
5160 h2 = h2+a(i1,i2)*a(n1+1,i2): NEXT
5170 a(i1,n2+2)=h2:h1=h1+h2*h2: NEXT:
     IF h1<1E-10 THEN 5220
5180 h2=h4/h1: FOR i2=1 TO n2:
     a(0,i2) = a(0,i2)+h2*a(n1+1,i2)
5190 REM PRINT#0,h5;
5200 NEXT: FOR i1=1 TO n1:
     a(i1,n2+1) = a(i1,n2+1)+h2*a(i1,n2+2)
5210 NEXT: NEXT
5220 RETURN
5230 REM--- Horner-Schema ---
5240 y = a(0,n2)
5250 FOR i2=n2-1 TO 1 STEP -1
5260 y = y*x+a(0,i2)
5270 NEXT i2
5280 RETURN
5290 :
5300 REM--- Graphik ausplotten ---
5310 CLS#0
5320 CALL &A000
5330 GOTO 2400
5340 :
5350 :
5360 :
5500 MEMORY &A000-1
5510 FOR a=&A000 TO &A0BF
5520 READ b$: POKE a,VAL("&"+b$)
5530 NEXT a
5540 :
5550 DATA cd,ba,bb,cd,e7,bb,32,bd,a0,cd,6c,a0
5560 DATA 21,8f,01,22,be,a0,11,00,00,3e,07,32
5570 DATA c0,a0,cd,7c,a0,0e,00,3a,c0,a0,47,e5
```

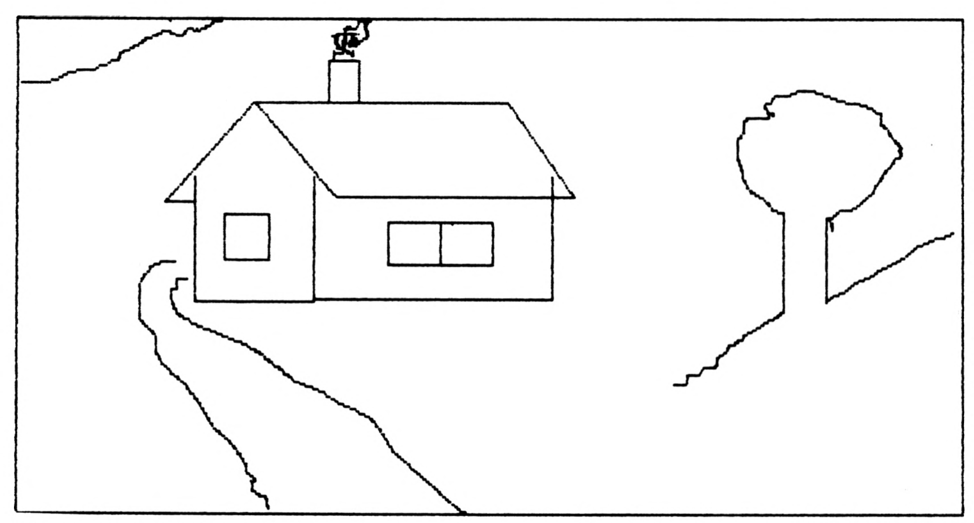
```
5580 DATA d5,c5,cd,f0,bb,c1,d1,21,bd,a0,be,e1
5590 DATA 37,20,01,a7,cb,11,2b,2b,10,e9,cd,af
5600 DATA a0,79,cd,a6,a0,13,e5,21,7f,02,37,ed
5610 DATA 52,e1,38,05,2a,be,a0,18,cc,23,7c,b5
5620 DATA c8,2b,11,00,00,22,be,a0,3e,07,bd,20
5630 DATA b9,7c,b4,20,b5,3e,04,32,c0,a0,18,ae
5640 DATA 3e,1b,cd,a6,a0,3e,33,cd,a6,a0,3e,15
5650 DATA cd,a6,a0,c9,e5,3e,42,cd,1e,bb,e1,28
5660 DATA 02,e1,c9,3e,0d,cd,a6,a0,3e,0a,cd,a6
5670 DATA a0,3e,1b,cd,a6,a0,3e,4c,cd,a6,a0,3e
5680 DATA 7f,cd,a6,a0,3e,02,cd,a6,a0,c9,cd,2e
5690 DATA bd,38,fb,cd,2b,bd,c9,3a,c0,a0,fe,07
5700 DATA c8,af,cb,11,cb,11,cb,11,c9,00,00,00
5710 RETURN
5720 REM--- Ende des Programmes ---
5730 END
```

16.4 Zeichnen mit dem Joystick

Durch die Möglichkeit einer komfortablen Joystickabfrage liegt es nahe, ein Programm zu entwickeln, das es ermöglicht, hochauflösende Zeichnungen mit Hilfe des Joysticks zu erzeugen. Der Joystick ist am USER PORT anzuschließen.

Das folgende lauffähige Teilprogramm benutzt die Variablen xp für die momentane x-Koordinate und yp als y-Koordinate. Der Bildausschnitt, in dem gezeichnet werden soll, ist durch i8,i9,j8,j9 bestimmt. Diese Bildbegrenzung erfolgt durch das Unterprogramm (Zeilen 340 bis 370), das keine Werte zuläßt, die die gegebenen Grenzen über- oder unterschreiten. Zeile 50 bestimmt den Bildschirmmittelpunkt als Anfangspunkt. Nach dem Einschalten des hochauflösenden Bildschirms (Zeile 120) wird der aktuelle Punkt P(xp,yp) markiert (Zeile 160). Wird der Joystick ausgelenkt, so erfolgt in der Zeile 170 ein Sprung zu Zeile 280. Hier wird entsprechend der Betätigungsrichtung zu der maßgebenden Zeile (Zeilen 490 bis 560) verzweigt. In diesen Zeilen wird je nach Joystick-Richtung der Wert dx und der Wert dy ggf. mit der positiven oder negativen Schrittweite d belegt. Der dx- bzw. dy-Wert wird in Zeile 330 zu dem xp- bzw. yp-Wert addiert. Nachdem eine eventuelle Grenzüberschreitung ausgeschlossen wurde (Zeile 340—370), wird an der neuermittelten Position ein Punkt gesetzt (Zeile 160).

Soll der „Cursor" bewegt werden, ohne zu zeichnen, so drücken wir auf die Feuertaste des Joystick. Die Abfrage a=JOY(0) (Zeile 170) liefert nun einen Wert a, der größer als 15 ist. In der Zeile 290 wird die p-Kennung auf 0 gesetzt (Stift hoch) und die Schrittweite d vergrößert. Mit diesem Programm wurde „von Hand" die Grafik 80 erzeugt.



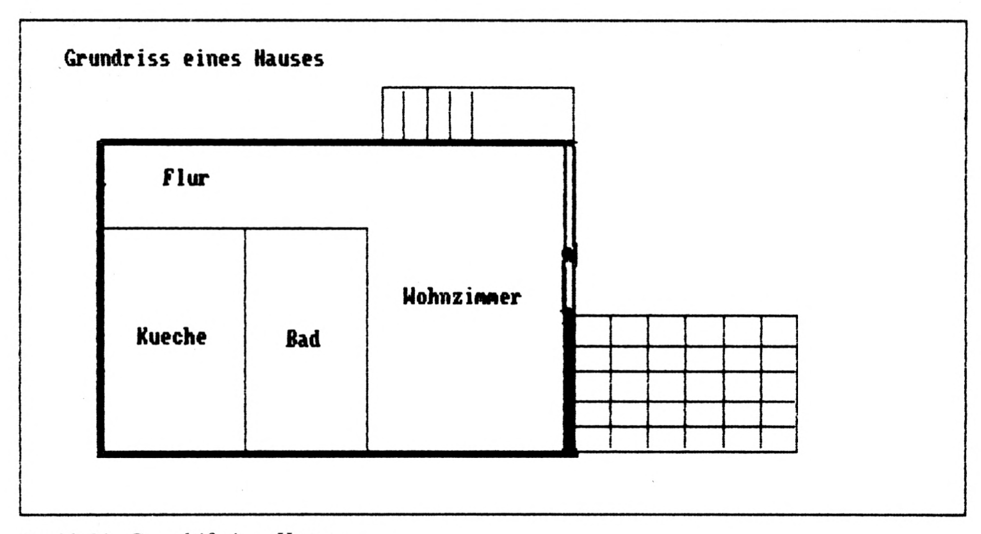
Grafik 80: Zeichnen mit dem Joystick

```
10 REM--- Zeichnen mit dem Joystick ---
20 i8 = 0: i9 = 639: REM-Bildschirm- -
30 j8 = 0: j9 = 399: REM-   pixels   -
50 xp = 320: yp = 200: d = 1: p = 0
120 MODE 2:BORDER 2:INK 0,1:INK 1,26
130 MOVE i8,j8: DRAW i9,j8, 1: DRAW i9,j9, 1
140 DRAW i8,j9, 1: DRAW i8,j8, 1
150 :
160 PLOT xq,yq, p: PLOT xp,yp, 1: xq=xp: yq=yp
170 a=JOY(0): IF a THEN 280
190 GOTO 170
200 :
280 IF a<16 THEN d=1 : p=1
290 IF a>15 THEN d=3 : p=0
300 ON a MOD 16 GOSUB
     490,500,170,510,520,530,170,540,550,560
310 :
330 xp=xp+dx: yp=yp+dy
340 IF yp<j8 THEN yp=j8
350 IF xp>i9 THEN xp=i9
360 IF xp<i8 THEN xp=i8
370 IF yp>j9 THEN yp=j9
380 GOTO 150
390 :
```

```
490 dx= 0:dy= d: RETURN
500 dx= 0:dy=-d: RETURN
510 dx=-d:dy= 0: RETURN
520 dx=-d:dy= d: RETURN
530 dx=-d:dy=-d: RETURN
540 dx= d:dy= 0: RETURN
550 dx= d:dy= d: RETURN
560 dx= d:dy=-d: RETURN
```

16.5 Mini-CAD-Programm

Das im Kapitel 16.4 beschriebene Programm zum Zeichnen mit dem Joystick wird nun zu einem kleinen CAD-Lehr-Programm erweitert. Die Leistungen sind ab der Zeile 1620 in PRINT-Anweisungen niedergelegt. In der Zeile 90 wird ein Befehlsstring b$ definiert. Jedes Zeichen von b$ soll eine bestimmte Leistung des Programmes aufrufen. Die Länge von b$ speichern wir in der Variablen lb ab.



Grafik 81: Grundriß eines Hauses

Wird eine Taste betätigt, so veranlaßt die Zeile 180 einen Sprung zu der Zeile 210. Hier wird der Befehlsring b$ nach dem getippten Zeichen a$ durchsucht. Wird in b$ das Zeichen a$ gefunden, so wird die Position in a1 gespeichert und die Schleife 210–230 verlassen. Gemäß der Position a1 wird nun in der Zeile 240 zu dem entsprechenden „Unterprogramm" verzweigt.

Drücken wir z. B. die Taste „3", so wird zur Zeile 600 verzweigt. Dort wird der „Zeichenstift" durch p=0 „hochgestellt" und der Löschmodus durch pp=0 ausgeschaltet. Durch die

Taste „3" werden die momentanen Cursor-Koordinaten xp,yp in den Variablen x1,y1 gespeichert. Bei jeder Cursor-Bewegung ändern sich die Koordinaten xp,yp. Die Koordinaten x1,y1 werden durch „3" geändert.

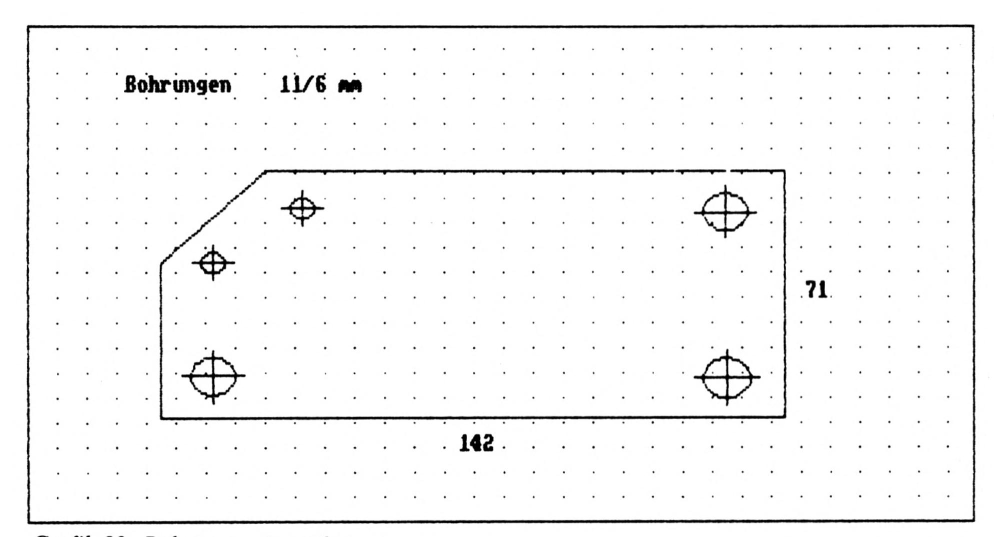
Drücken wir die Taste „x", so wird zu BASIC verzweigt (Zeile 1300) und das Programm beendet.

Drücken wir die Taste „2", so wird in der Zeile 1030 der „Zeichenstift" gesenkt (p=1, Zeile 160). Die nachfolgenden Joystick-Bewegungen zeichnen Linien auf den Bildschirm.

Zum Zeichnen von aufeinanderfolgenden Geraden kann man wie folgt vorgehen: Zunächst wird mit „3" die aktuelle Cursor-Position in x1,y1 gespeichert. Mit dem Joystick gehen wir nun zu einem anderen Bildpunkt. Hierbei wird keine Linie gezeichnet. Drücken wir dann die Taste „4" (s. Zeile 630), so wird eine Gerade vom markierten Punkt P1 zur aktuellen Cursor-Position gezogen. Durch die Zeile 680 wird der Endpunkt der gezeichneten Geraden automatisch zum markierten Anfangspunkt P1. Dadurch können leicht fortlaufende Linienzüge gezeichnet werden. Soll der Punkt P1 nicht nachgeführt werden, so ist die Zeile 680 zu neutralisieren. In diesem Falle wird P1 nur durch die Taste „3" neu markiert, und es lassen sich leicht „sternförmige" Geradenbüschel zeichnen.

Ist ein Punkt P1 markiert, so kann mit dem Befehl „r" ein „R"echteck gezeichnet werden. Das Rechteck ist durch die diagonal gegenüberliegenden Eckpunkte bestimmt, die durch die momentane Cursor-Position und den markierten Punkt P1 gegeben sind. Auf diese Weise wurde die Grafik 81 erzeugt.

In vielen Fällen muß ein Hilfsraster (Grafik 82) auf dem Bildschirm sichtbar sein, damit maßstäblich gezeichnet werden kann. Der Befehl „g" zeichnet „G"itterpunkte (Zeilen 1320–1410). Wird erneut „g" eingegeben, so werden die Gitterpunkte gelöscht.



Grafik 82: Bohrungen einer Platine

Zum Zeichnen von Kreisen markieren wir mit „3“ den Kreismittelpunkt und bewegen den Cursor mit dem Joystick zu einem gewünschten Randpunkt des Kreises. Drücken wir nun auf die Taste „k“, so wird ein „K“reis gezeichnet (Zeilen 710–990). Auf diese Weise wurde die Grafik 82 hergestellt.

Durch den Befehl „l“ ist es möglich, Figuren zu „l“öschen. Hierbei wird l einfach vor dem jeweils auszuführenden Befehl eingegeben. Soll z. B. ein Kreis gelöscht werden, so ist der Mittelpunkt zu markieren, der Cursor auf die Kreislinie zu stellen und „l“ gefolgt von „k“ einzugeben („l“ösche „K“reis).

Soll eine Gerade gelöscht werden, so ist der eine Geradenendpunkt mit „3“ zu speichern, der Cursor auf den zweiten Linienendpunkt zu stellen und „l“ gefolgt von „4“ zu drücken. Analog kann ein gezeichnetes Rechteck mit „l“ „r“ gelöscht werden. Weil eine Linie aus vielen Bildschirmpixeln besteht, kann es vorkommen, daß einzelne Punkte nicht gelöscht werden. Solche verstreuten Pixel können leicht gelöscht werden, indem man mit dem Cursor zur Löschstelle fährt. Nach dem Drücken der Taste „1“ können die verstreuten Pixel dann mit der Leertaste gelöscht werden.

Für die Beschriftung der Zeichnung ist die Taste „1“ vorgesehen, die in den Schreibmodus umschaltet (Zeile 1440 ff.). Nach dem Drücken von „1“ erscheint jedes getippte Zeichen auf dem Bildschirm. Drücken wir auf ENTER, so springt der Cursor in die nächste Schreibzeile (s. Zeile 1520). Durch die Taste mit dem nach oben gerichteten Pfeil springt der Cursor an den Anfang der darüberliegenden Zeile (s. 1530). Die Leertaste „radiert“ von „links nach rechts“ (Zeile 1550) und die DEL-Taste von „rechts nach links“ (Zeile 1540). Bewegen wir den Joystick, so wird der Schreibmodus verlassen.

Die Ausgabe der Grafik auf dem „D“rucker erfolgt mit dem Befehl „d“ (Zeile 1260).

```
10 REM !-----------------------------------!
15 REM ! Ein Mini-CAD-Programm zum Zeichnen !
20 REM ! und Loeschen von Kurven, Geraden,  !
22 REM ! Kreisen, Rechtecken  und zum       !
25 REM ! Schreiben von Texten in die Grafik !
30 REM !-----------------------------------!
32 GOSUB 3000
35 GOTO 1600
40 :
50 :
60 i8 = 0:i9=638: REM--- Bildschirm- ---
70 j8 = 0:j9=399: REM--- pixels      ---
80 xp=320: yp=200: xl=xp: yl=yp: d=1: nn=99
90 b$="klrxgdl234": REM--- Befehlsstring---
95 lb = LEN(b$)
110 :
130 MOVE i8,j8:DRAW i9,j8,1:DRAW i9,j9,1
```

```
140 DRAW i8,j9,1:DRAW i8,j8,1
145 :
150 IF pp=3 THEN PLOT xp,yp,0
155 IF TEST(xp,yp) THEN 170
160 PLOT xq,yq,p: PLOT xp,yp,1: xq=xp: yq=yp
170 a = JOY(0): IF a THEN 280
180 a$=INKEY$: IF a$<>"" THEN 210
190 GOTO 170
200 :
210 FOR i=1 TO lb
220 IF a$=MID$(b$,i,1) THEN al=i:i=lb
230 NEXT
235 :
240 ON al GOTO 710,1080,1100,1300,1320,1260,
      1440,1030,600,630,260
260 GOTO 150
270 :
280 IF a<16 THEN FOR i=0 TO nn:NEXT: d=1
290 IF a>16 THEN d=3
300 ON a MOD 16 GOSUB 490,500,170,510,520,
    530,170,540,550,560
310 :
330 xp=xp+dx: yp=yp+dy
340 IF yp<j8 THEN yp=j8
350 IF xp>i9 THEN xp=i9
360 IF xp<i8 THEN xp=i8
370 IF yp>j9 THEN yp=j9
380 GOTO 150
390 :
490 dx= 0:dy= d: RETURN
500 dx= 0:dy=-d: RETURN
510 dx=-d:dy= 0: RETURN
520 dx=-d:dy= d: RETURN
530 dx=-d:dy=-d: RETURN
540 dx= d:dy= 0: RETURN
550 dx= d:dy= d: RETURN
560 dx= d:dy=-d: RETURN
570 :
600 REM--- markieren eines Punktes p1 ---
610 p=0: pp=0: x1=xp: y1=yp
615 al=11: GOTO 150
620 :
```

```
630 REM- Gerade zum markierten Punkt zeichnen -
640 p=1: IF pp THEN p=0
650 IF pp THEN MOVE x1+1,y1: DRAW xp+1,yp,p
660 IF pp THEN MOVE x1,y1-1: DRAW xp,yp-1,p
670 MOVE x1,y1: DRAW xp,yp,p: p=0: pp=0
680 x1=xp: y1=yp
690 a1=11: GOTO 150
710 REM--- Kreis um markierten     ---
715 REM--- Mittelpunkt m zeichnen ---
720 r=SQR((xp-x1)*(xp-x1)+(yp-y1)*(yp-y1))
730 DEG: h=r+6
740 IF x1-h<i8 OR y1-h<j8 THEN 980
750 IF pp=1 THEN 840
760 FOR a=0 TO 360 STEP 10
770 a2=x1+r*COS(a): b2=y1+r*SIN(a)
780 IF a>0 THEN MOVE a1,b1: DRAW a2,b2,1
790 a1=a2: b1=b2
795 NEXT a
800 PLOT x1,y1,1
810 MOVE x1-h,y1: DRAW x1+h,y1,1
820 MOVE x1,y1-h: DRAW x1,y1+h,1
830 GOTO 980
835 :
840 h=h+3: rr=r+1
850 FOR i=-1 TO 1
860 FOR a=0 TO 360 STEP 10
870 a2=x1+rr*COS(a): b2=y1+i+rr*SIN(a)
880 IF a>0 THEN MOVE a1,b1: DRAW a2,b2,0
890 a1=a2: b1=b2
895 NEXT a
900 FOR a=0 TO 360 STEP 10
910 a2=x1+i+rr*COS(a): b2=y1+rr*SIN(a)
920 IF a>0 THEN MOVE a1,b1: DRAW a2,b2,0
930 a1=a2: b1=b2
935 NEXT a
940 MOVE x1-h,y1+i: DRAW x1+h,y1+i,0
950 MOVE x1+i,y1-h: DRAW x1+i,y1+h,0
960 NEXT i
970 REM--- PLOT x1,y1,0
975 :
980 RAD: p=0: pp=0: xp=x1: yp=y1
990 a1=11: GOTO 150
```

```
1000 :
1010 :
1020 REM--- mit CURSOR zeichnen/loeschen ---
1030 d=1:  IF pp>1 THEN pp=0: p=0: GOTO 1060
1040 nn=0: IF pp=1 THEN pp=3
1050 p=0:  IF pp=0 THEN p=1: pp=2
1060 al=11: GOTO 150
1070 :
1080 REM---  Loeschen ---
1085 p=0: pp=1: d=1: al=11:   GOTO 150
1090 :
1100 REM--- Rechteck pl zu CURSOR ---
1110 xq=xl: yq=yl
1120 IF xp<xq THEN xq=xp: xp=xl
1130 IF yp<yq THEN yq=yp: yp=yl
1140 IF pp THEN 1180
1150 MOVE xq,yq: DRAW xp,yq,1
1160 DRAW xp,yp, 1: DRAW xq,yp, 1
1170 DRAW xq,yq,1: GOTO 1230
1175 :
1180 FOR i=-2 TO 2
1190 MOVE xq+i,yq+i: DRAW xp+1,yq,0
1200 DRAW xp+1,yp+1, 0: DRAW xq,yp+1, 0
1210 DRAW xq+i,yq+i,0
1220 NEXT i: p=0: pp=0
1230 xp=xl: yp=yl
1240 al=11: GOTO 150
1250 :
1260 REM--- Hardcopy ---
1270 CALL &A000
1280 al=11: GOTO 150
1290 :
1300 END:          REM--- zurueck zu BASIC ---
1310 :
1320 REM--- Gitterpunkte zeichnen/loeschen ---
1330 FOR j=j8 TO j9 STEP (j9-j8)/19.9
1340 :  FOR i=i8 TO i9 STEP (i9-i8)/31.9
1350 :      PLOT i,j,p
1360 :  NEXT i
1370 NEXT j
1380 MOVE i8,j8:DRAW i9,j8,1:DRAW i9,j9,1
1390 DRAW i8,j9,1:DRAW i8,j8,1
```

```
1400 p=0
1410 al=11: GOTO 150
1420 :
1430 :
1440 REM--- Text in die Grafik schreiben ---
1450 TAG
1460 xq=xp: yq=yp
1470 PLOT xp,yp,1
1480 IF JOY(0)
     THEN PLOT xp,yp,0: TAGOFF: GOTO 150
1490 a$=INKEY$: IF a$="" THEN 1480
1500 PLOT xp,yp,0: PLOT 0,0,1
1510 IF a$="3" THEN TAGOFF: GOTO 210
1520 IF a$=CHR$(13)
     THEN yp=yp-16: xp=xq: GOTO 1460
1530 IF a$=CHR$(94) OR a$=CHR$(240)
     THEN yp=yp+16: xp=xq: GOTO 1460
1540 IF a$=CHR$(127) THEN
     xp=xp-8: MOVE xp,yp: PRINT " ";: GOTO 1580
1550 IF a$=CHR$(32) THEN
     MOVE xp,yp: PRINT " ";: GOTO 1570
1560 MOVE xp,yp: PRINT a$;
1570 xp=xp+8
1580 GOTO 1470
1590 :
1600 MODE 2:BORDER 2:INK 0,1:INK 1,26
1610 LOCATE 20,1
1620 PRINT "Befehlssatz fuer das  ";
1630 PRINT "Mini-CAD-Programm":PRINT
1640 PRINT
1650 PRINT " ´CURSOR´ wird durch Joystick ";
1660 PRINT "bewegt         g  punktiert ein ";
1670 PRINT "Gitternetz"
1680 PRINT "    langsam (Joy ohne Feuertaste)";
1690 PRINT "                 Eingabe <l><g> ";
1700 PRINT "loescht das Gitter";
1710 PRINT "    schnell (Joy mit  Feuertaste)"
1720 PRINT SPC(43);"k  zieht Kreis um pl ";
1730 PRINT "(mit 3 markiert)";
1740 PRINT " l  fuer Texteingabe             ";
1750 PRINT "                 Radius entspricht ";
1760 PRINT "´CURSOR´-"
```

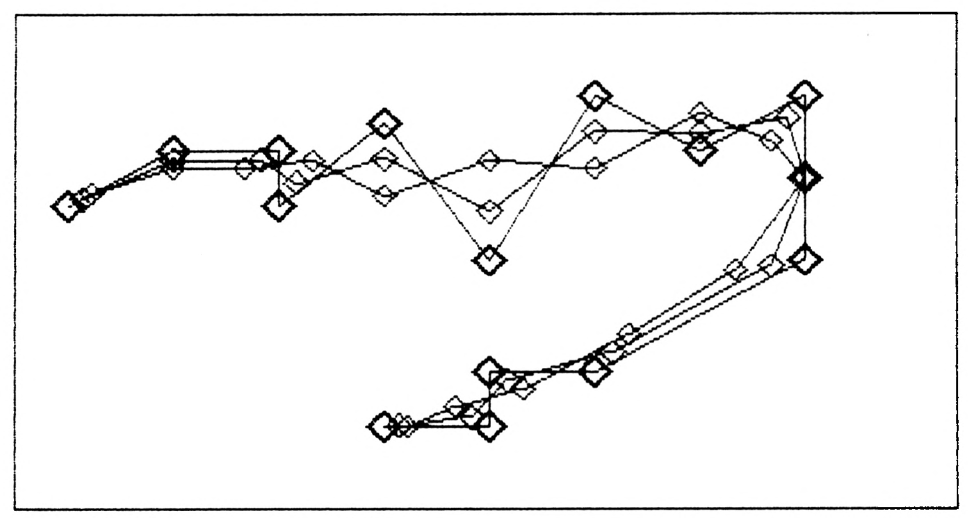
```
1770 PRINT "    <ENTER> geht auf die neue ";
1780 PRINT "Zeile";
1790 PRINT "              Stellung"
1800 PRINT "    <^>      eine Zeile hoch      ";
1810 PRINT "              <1><k> loescht den ";
1820 PRINT "Kreis, wenn"
1830 PRINT "    loeschen mit <DEL> oder ";
1840 PRINT "<BLANK>              ";
1850 PRINT "der ´CURSOR´ auf dem Kreisrand"
1860 PRINT "    <3> beendet Texteingabe      ";
1870 PRINT "              steht und der ";
1880 PRINT "Mittelpunkt pl"
1890 PRINT SPC(46);"markiert ist"
1900 PRINT " 2  Joy zieht Kurven, ausschalt. ";
1910 PRINT "<2>"
1920 PRINT "    <1><2>  loescht beim Zeichnen";
1930 PRINT "           l  Loeschen (pl muss ";
1940 PRINT "markiert sein)"
1950 PRINT SPC(46);"z.B. eine Gerade wird ";
1960 PRINT "geloescht:"
1970 PRINT " 3  Punkt pl markieren (pl ";
1980 PRINT "merken) !!!         ";
1990 PRINT "<3> beim Geradenanfang, dann mit"
2000 PRINT "    abschalten von 1 und 2      ";
2010 PRINT "               ´CURSOR´ zum ";
2020 PRINT "Endpunkt, <1><4>"
2030 PRINT
2040 PRINT " 4  Gerade von pl zum ´CURSOR´ ";
2050 PRINT "ziehen       ";
2060 PRINT "r  Rechteck von pl zum ´CURSOR´"
2070 PRINT
2080 PRINT " x  zurueck zu BASIC              ";
2090 PRINT "           d  Grafik auf dem ";
2100 PRINT "Drucker ausgeben"
2110 LOCATE 27,25
2120 PRINT "<TASTE>-druecken fuer Start"
2130 a$=INKEY$: IF a$="" THEN 2130
2140 CLS
2150 RUN 40
2160 :
3000 MEMORY &A000-1
3010 FOR a=&A000 TO &A0BF
```

```
3020 READ b$: POKE a,VAL("&"+b$)
3030 NEXT a
3040 :
3050 DATA cd,ba,bb,cd,e7,bb,32,bd,a0,cd,6c,a0
3060 DATA 21,8f,01,22,be,a0,11,00,00,3e,07,32
3070 DATA c0,a0,cd,7c,a0,0e,00,3a,c0,a0,47,e5
3080 DATA d5,c5,cd,f0,bb,c1,d1,21,bd,a0,be,e1
3090 DATA 37,20,01,a7,cb,11,2b,2b,10,e9,cd,af
3100 DATA a0,79,cd,a6,a0,13,e5,21,7f,02,37,ed
3110 DATA 52,e1,38,05,2a,be,a0,18,cc,23,7c,b5
3120 DATA c8,2b,11,00,00,22,be,a0,3e,07,bd,20
3130 DATA b9,7c,b4,20,b5,3e,04,32,c0,a0,18,ae
3140 DATA 3e,1b,cd,a6,a0,3e,33,cd,a6,a0,3e,15
3150 DATA cd,a6,a0,c9,e5,3e,42,cd,1e,bb,e1,28
3160 DATA 02,e1,c9,3e,0d,cd,a6,a0,3e,0a,cd,a6
3170 DATA a0,3e,1b,cd,a6,a0,3e,4c,cd,a6,a0,3e
3180 DATA 7f,cd,a6,a0,3e,02,cd,a6,a0,c9,cd,2e
3190 DATA bd,38,fb,cd,2b,bd,c9,3a,c0,a0,fe,07
3200 DATA c8,af,cb,11,cb,11,cb,11,c9,00,00,00
3210 RETURN
```

16.6 Glätten von Meßreihen

Bei naturwissenschaftlichen Experimenten fallen gewöhnlich Meßwerte an, die infolge von Meßfehlern Ungenauigkeiten aufweisen. Wird z. B. ein Vielkanalanalysator verwendet, so können die benachbarten Kanalwerte stark streuen. Es besteht die Möglichkeit, diese „verrauschten“ Meßreihen durch die Bildung eines Mittelwertes zwischen den benachbarten Meßpunkten zu glätten. Hierbei kann der Einfluß der Nachbarwerte auf den Mittelwert unterschiedlich stark berücksichtigt werden. Wir setzen voraus, daß sich die n-Punkte P1(u1,v1),P2(u2,v2),. . .,Pn(un,vn) der Meßreihe fortlaufend, im Sinne aufsteigender Indizes, aus dem Experiment ergeben.

In der Grafik 83 sind die Meßwerte durch große Doppelrechtecke markiert. Wir wollen nun die gemittelten Meßwerte (kleine Rechtecke in der Grafik 83) bestimmen. Hierzu betrachten wir die Abb. 25, in der die 3 Punkte P1(u1,v1),P2(u2,v2),P3(u3,v3) stellvertretend für 3 beliebig aufeinanderfolgende Meßpunkte aufgetragen sind. Um die Schwankungen der Linien P1-P2-P3 zu mildern, wollen wir P2 durch einen gemittelten Punkt Q2 ersetzen. Der Punkt Q2 soll auf der Seitenhalbierenden liegen, die in der Abb. 25 strichpunktiert gezeichnet ist.



Grafik 83: Glättung der x *und* y*-Meßwerte*

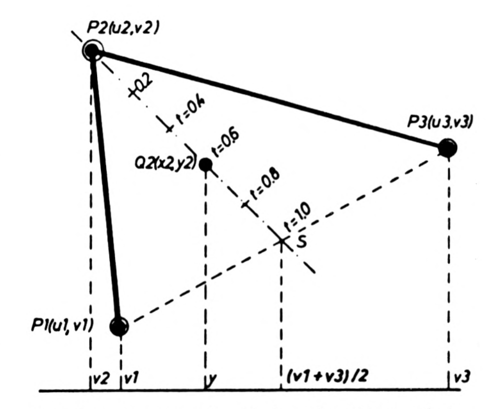


Abb. 25: Ersetzen des Meßpunktes P2 *durch den gemittelten Punkt* Q2

Der Parameter t soll den Einfluß der Nachbarpunkte auf den gemittelten Punkt Q2 beschreiben. Für t=0 haben die Nachbarpunkte keinen Einfluß, und es gilt Q2=P2. Für t=1 ergibt sich Q2 als arithmetisches Mittel von P1 und P3, d. h. P2 hat keinen Einfluß auf Q2. Für t=2/3 ist Q2 der Schwerpunkt des Dreiecks P1-P2-P3. Die Koordinaten des Punktes Q2 seien x2,y2. Nach dem Strahlensatz (s. Abb. 25) gilt

$$(y2 - v2)/t = (v1 + v3)/2 - v2$$

oder umgestellt

$$y2 = v2 + t*((v1 + v3)/2 - v2),$$

und entsprechend ist

$$x2 = u2 + t*((u1 + u3)/2 - u2).$$

Dabei haben wir die Koordinaten x2,y2 des gemittelten Punktes Q2 unter Berücksichtigung der Nachbarpunkte von P2 ausgedrückt. Diese Zeilen gelten analog für den k-ten Punkt (Zeilen 1000,1020). Der erste und letzte Punkt der Meßreihe erfordern eine Sonderbehand-

lung. Hierzu wird in den Zeilen 910 bis 940 der Vorgängerpunkt u(0),v(0) und der Nachfolgepunkt u(n+1),v(n+1) zusätzlich zu den vorgegebenen n-Meßpunkten P1,P2,. . .,Pn besetzt.

In dem folgenden Testprogramm sind 15 Meßpunkte P1,P2,. . .,Pn in den Data-Zeilen 220,240,250 abgelegt, die in die Felder u(),v() eingelesen werden (Zeile 390). Nachdem der Zeichenmaßstab mx,my (Zeile 440) berechnet wurde, wird der hochauflösende Bildschirm eingeschaltet (Zeile 480).

Danach werden für t=0 (Zeile 540) die gemittelten Punkte x(),y() berechnet. Wegen t=0 (s. Zeilen 910 bis 1020) entfällt der Einfluß der Nachbarpunkte, d. h. für t=0 werden die Original-Meßpunkte (Zeilen 660,680) in die Bildpunkte (i2,j2) umgerechnet. Diese Punkte werden durch Linien (Zeile 700) verbunden. Die Zeilen 720 bis 740 dienen lediglich zur Kennzeichnung der gemittelten Punkte durch kleine Rechtecke. Für t=0 werden die vorgegebenen Meßpunkte zusätzlich durch größere Rechtecke gekennzeichnet.

Gilt $0<t<1$, so wird in dem Unterprogramm (840 bis 1060) eine Mittellung durchgeführt. Die Meßreihe wird geglättet. Abgesehen vom ersten und n-ten Punkt werden bei der Glättung jeweils 3 aufeinanderfolgende Meßwerte zur Berechnung der gemittelten Werte verwendet. Diese gemittelten Meßpunkte werden durch Geraden verbunden. Sollen diese gemittelten Punkte durch stetige Kurven verbunden werden, so ist hierfür z. B. der Algorithmus des Kapitels 14.2 geeignet.

Bei den bisherigen Betrachtungen wurden sowohl die x-Werte als auch die y-Werte geglättet. Für zahlreiche Anwendungen ist lediglich eine Glättung der y-Werte durchzuführen. In diesem Fall ist die Zeile 1000 des folgenden Programmes durch ein hinter die Zeilennummer eingefügtes „REM“ zu neutralisieren. Als Ergebnis einer y-Glättung ergibt sich die Grafik 84. Ist dagegen nur die Zeile 1000 aktiv und die Zeile 1020 neutralisiert, so ergibt sich eine Glättung der x-Werte (Grafik 85).

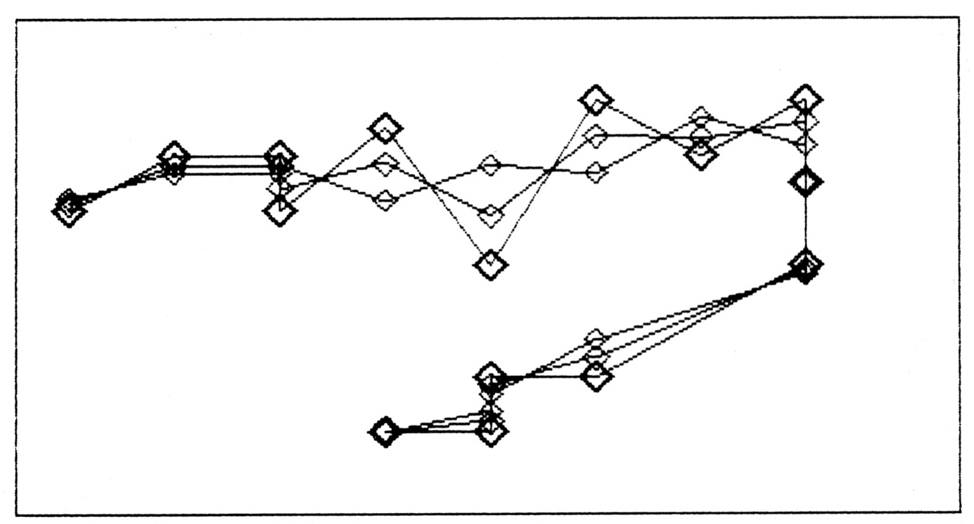
Das Programm zum Glätten von Meßwerten kann in vielfacher Weise zur Herstellung von künstlerischen Grafiken verwendet werden. Hierbei sind lediglich geeignete Punkte in den Zeilen 220,240,250 einzugeben. Die Reihenfolge der Punkte entspricht der Reihenfolge der gezeichneten Linien. Durch die Schleife in Zeile 540 mit geeigneten Grenzen und kleiner Schrittweite ergeben sich durch den Unterprogrammaufruf in der Zeile 600 jeweils geänderte Punkte, die dann entsprechend verbunden werden. Wird die Zeile 540 durch

```
540 FOR t=-.2 TO 1.1 STEP .08
```

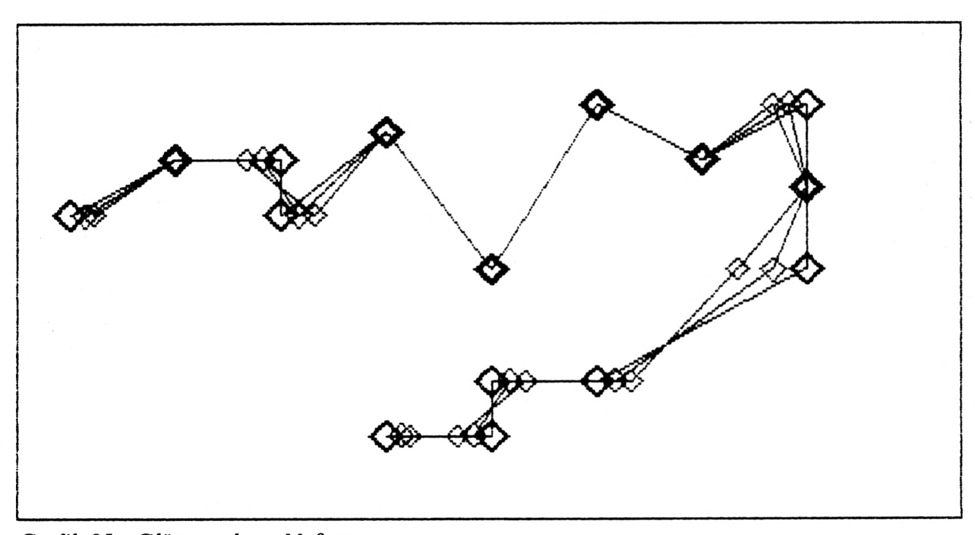
und die Zeile 220 durch

```
220 DATA 5, -.5,-3, -3,3, 0,1, 3,3, .5,-3
```

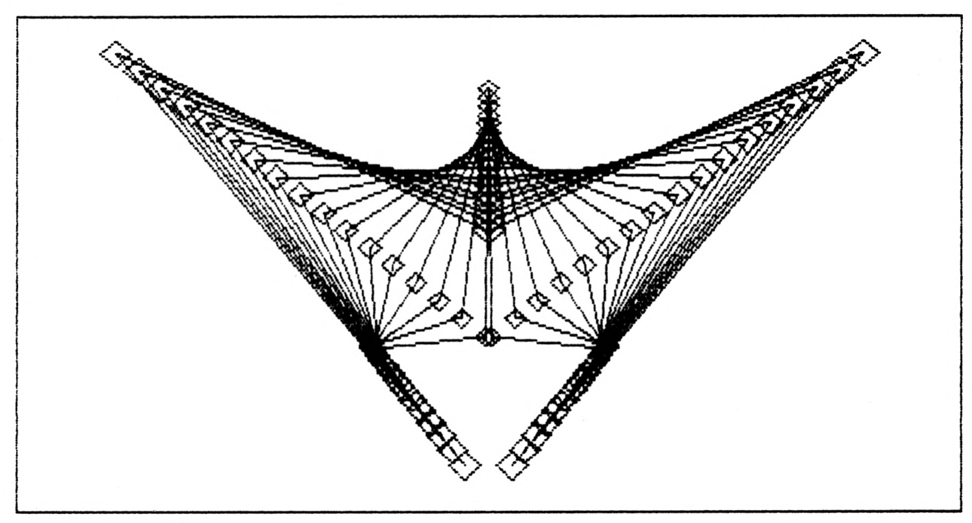
ersetzt, so ergibt sich die Grafik 86 für eine x- und y-Glättung, Grafik 87 für eine y-Glättung (Zeile 1000 neutralisiert) und Grafik 88 für eine x-Glättung (Zeile 1020 neutralisieren).



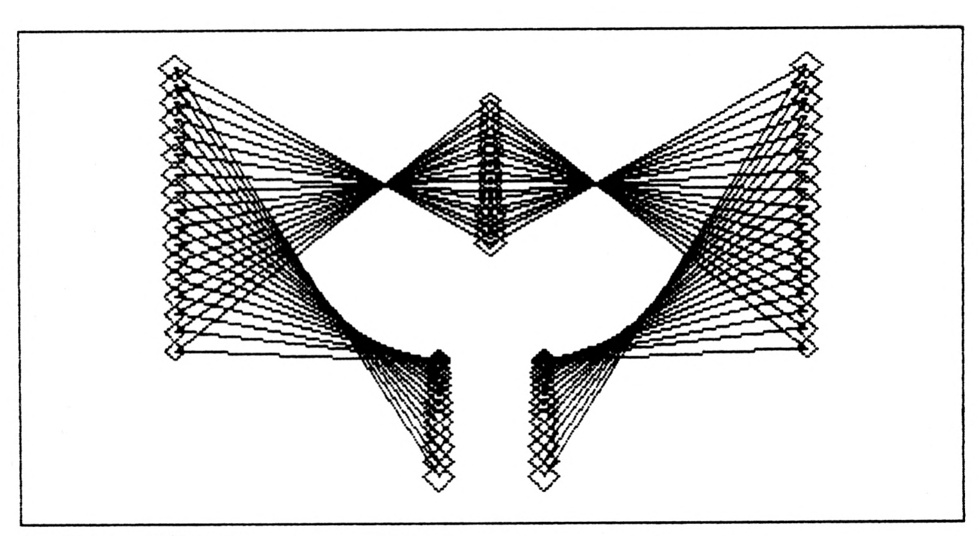
Grafik 84: Glättung der y-Meßwerte



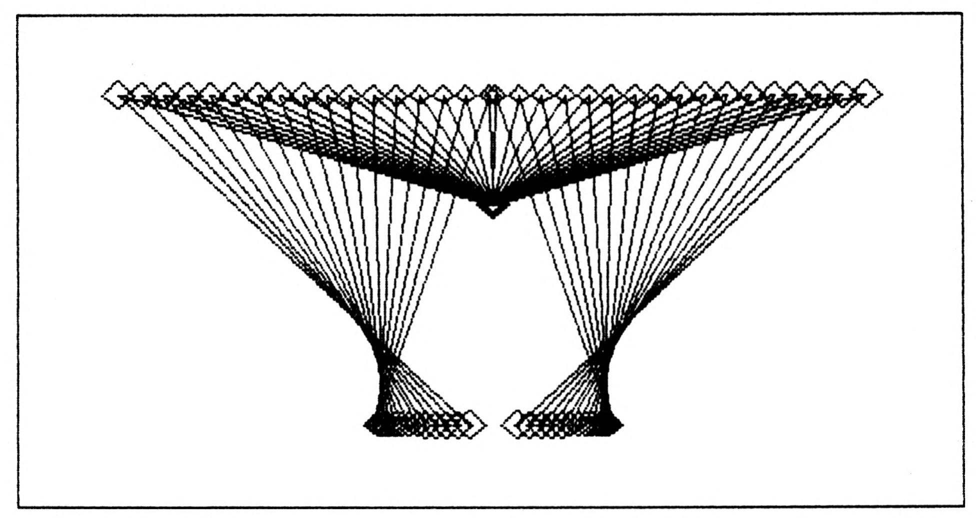
Grafik 85: Glättung der x-Meßwerte



Grafik 86: x,y-*Glättung*



Grafik 87: y-*Glättung*



Grafik 88: x-*Glättung*

```
160 REM Glaetten von Messpunkten p1,p2,...,pn
180 :
200 REM   n, u1,v1, u2,v2, ........, un,vn
220 DATA 15, -4,1, -3,2, -2,2, -2,1, -1,2.5
240 DATA      0,0,  1,3,  2,2,  3,3,  3,1.5
250 DATA      3,0,  1,-2, 0,-2, 0,-3, -1,-3
260 :
280 DIM u(100), v(100), x(100), y(100)
300 x8 =-4.5: x9 =  4.5: y8 =-4.5: y9 =  4.5
320 i8 = 0.5: i9 =639.5: j8 = 0.5: j9 =399.5
340 :
360 READ n
380 FOR k=1 TO n
390 READ u(k),v(k): x(k)=u(k): y(k)=v(k)
400 NEXT k
410 :
440 mx = (i9-i8)/(x9-x8): my = (j9-j8)/(y9-y8)
450 :
480 MODE 2:BORDER 2:INK 0,1: INK 1,24
490 MOVE 0,0: DRAW 639,0,1: DRAW 639,399,1
500 DRAW 0,399,1: DRAW 0,0,1
520 :
540 FOR t=0 TO 0.67 STEP 0.33
600 :  GOSUB 840
```

```
620 :
640 :   FOR k=1 TO n
660 :      i2 = i8 + mx * (x(k) - x8)
680 :      j2 = j8 + my * (y(k) - y8)
700 :      IF k>1
           THEN MOVE i1,j1: DRAW i2,j2, 1
710 :      MOVE i2-(10-3*t), j2
720 :      DRAW i2, j2+(10-3*t), 1
725 :      DRAW i2+(10-3*t), j2, 1
730 :      DRAW i2, j2-(10-3*t), 1
735 :      DRAW i2-(10-3*t), j2, 1
740 :      IF t=0 THEN MOVE i2-12, j2:
           DRAW i2, j2+12, 1:DRAW i2+12, j2, 1:
           DRAW i2, j2-12, 1:DRAW i2-12, j2, 1
760 :      i1 = i2: j1 = j2
780 :   NEXT k
800 NEXT t
810 :
820 GOTO 1100
830 :
840 REM !--------------------------------------!
850 REM ! geglaettete Punkte mit den Koordinaten!
860 REM ! x(), y() berechnen.Eingangswerte sind !
870 REM ! u(), v() und der Glaettungsparameter  !
880 REM ! t mit 0 <= t <= 1,  z.B.  t = 0.667   !
890 REM !--------------------------------------!
900 :
910 u(0) = u(1) + t*( u(1) - u(2) )/2
920 v(0) = v(1) + t*( v(1) - v(2) )/2
930 u(n+1) = u(n) + t*( u(n) - u(n-1) )/2
940 v(n+1) = v(n) + t*( v(n) - v(n-1) )/2
960 :
980 FOR k=1 TO n
1000 :  x(k) = u(k)+t*((u(k-1)+u(k+1))/2-u(k))
1020 :  y(k) = v(k)+t*((v(k-1)+v(k+1))/2-v(k))
1040 NEXT k
1060 RETURN
1080 :
1100 a$=INKEY$: IF a$="" THEN 1100
```

17. Hardcopy-Routine

Um die erstellten Grafiken „schwarz auf weiß“ vor sich haben zu können, ist die Ausgabe des Bildschirminhaltes auf einem Drucker nötig. Das folgende Programm ermöglicht eine Hardcopy des Bildschirmes auf einem SCHNEIDER-Drucker. Tippen Sie das Programm sehr sorgfältig ab, und sichern Sie es vor dem Start. Wenn Sie eine Hardcopy erstellen wollen, so starten Sie zunächst diese Hardcopy-Routine. Danach laden Sie das Programm zur Herstellung der Grafik und bauen einen CALL &A000 ein. Dieser „Einbau“ sieht z. B. so aus:

```
1670 a$=INKEY$: IF a$="" THEN 1670
1680 IF a$="d" THEN CALL &A000
```

Dabei stehen die beiden Programmzeilen am Programmende. Wird nach Erstellung der Grafik die Taste „d“ gedrückt, so wird die Hardcopy-Routine aufgerufen und der Bildschirminhalt auf dem Drucker ausgegeben. Nach Beendigung der Hardcopy meldet sich der Rechner mit READY. Nun kann durch nochmaliges Starten des Programmes eine weitere Hardcopy erstellt werden, oder es wird ein neues Programm geladen. In dieses Programm wird wieder ein CALL &A000 eingefügt, und es kann eine Bildschirm-Hardcopy hergestellt werden.

```
6000 MEMORY &A000-1
6010 FOR a=&A000 TO &A0BF
6020 READ b$: POKE a,VAL("&"+b$)
6030 NEXT a
6040 :
6050 DATA cd,ba,bb,cd,e7,bb,32,bd,a0,cd,6c,a0
6060 DATA 21,8f,01,22,be,a0,11,00,00,3e,07,32
6070 DATA c0,a0,cd,7c,a0,0e,00,3a,c0,a0,47,e5
6080 DATA d5,c5,cd,f0,bb,c1,d1,21,bd,a0,be,e1
6090 DATA 37,20,01,a7,cb,11,2b,2b,10,e9,cd,af
6100 DATA a0,79,cd,a6,a0,13,e5,21,7f,02,37,ed
6110 DATA 52,e1,38,05,2a,be,a0,18,cc,23,7c,b5
6120 DATA c8,2b,11,00,00,22,be,a0,3e,07,bd,20
6130 DATA b9,7c,b4,20,b5,3e,04,32,c0,a0,18,ae
6140 DATA 3e,1b,cd,a6,a0,3e,33,cd,a6,a0,3e,15
6150 DATA cd,a6,a0,c9,e5,3e,42,cd,1e,bb,e1,28
6160 DATA 02,e1,c9,3e,0d,cd,a6,a0,3e,0a,cd,a6
6170 DATA a0,3e,1b,cd,a6,a0,3e,4c,cd,a6,a0,3e
6180 DATA 7f,cd,a6,a0,3e,02,cd,a6,a0,c9,cd,2e
6190 DATA bd,38,fb,cd,2b,bd,c9,3a,c0,a0,fe,07
6200 DATA c8,af,cb,11,cb,11,cb,11,c9,00,00,00
```

18. Befehlsliste SCHNEIDER-BASIC

Bei der hier aufgeführten Kurzübersicht handelt es sich um eine Zusammenstellung der wichtigsten Befehle des SCHNEIDER-BASIC. Nicht behandelt werden die Befehle zur Musikerzeugung. Aufgeführt werden jeweils nur die Befehlssyntax sowie die Beschreibung der Wirkung.

Programmierhilfen

AUTO N,M	Automatische Zeilennumerierung ab Zeile N mit der Schrittweite M.
RENUM N,A,M	Umnumerieren der Zeilennummern. N entspricht der ersten gewünschten Zeilennummer. A gibt an, ab wo innerhalb des Programmes umnumeriert werden soll. M entspricht der Schrittweite. Alle Sprungbefehle werden aktualisiert.
RESTORE (Zeilennummer)	Setzt den Data-Zeiger auf den ersten im Programm auftretenden Wert oder auf den ersten Wert der angegebenen „Data"-Zeilennummer.
MERGE "Prog.-Name"	Anhängen eines Programmes an das im Speicher befindliche.
TRON TROFF	Nach dem Kommando wird vor Ausführung einer Zeile die jeweilige Zeilennummer in eckigen Klammern auf dem Bildschirm ausgegeben. TROFF beendet den TRON-Befehl.

Grafik-Befehle

MODE 0	Vielfarb-Auflösung 20 Spalten oder waagerechte Zeichen 16 verschiedene Farben darstellbar	160 x 200 Punkte
MODE 1	Normal-Auflösung 40 Spalten 4 verschiedene Farben darstellbar	320 x 200 Punkte
MODE 2	Hoch-Auflösung 80 Spalten 2 verschiedene Farben darstellbar	640 x 200 Punkte

In allen drei Bildschirmmodi gibt es 25 Zeilen (senkrechte Zeichen). Auch die Koordinatenangaben für die Bildschirmpixel sind in allen drei Modi die gleichen.

`INK N,C1[,C2]`	Durch das INK-Kommando wird (werden) dem Farbstift F die Farbe(n) C1 (und C2) zugeordnet.
`PEN F`	Ordnet dem Schreibstift den angegebenen Farbstift F zu.
`PAPER F`	Einschalten des Farbstiftes F zum Ausfüllen des beschreibbaren Hintergrundes.
`BORDER C1[,C2]`	Einschalten der Rahmenfarbe C1 bzw. der Farben C1 und C2, die dann blinken. Als Rahmenfarbe ist jede der 27 Farben möglich.
`SPEED INK N,M`	Setzt die Länge der Farbdauer bei Farbwechseln (Blinken) für INK und BORDER. N entspricht der Dauer der ersten Farbe. M entspricht der Dauer der zweiten Farbe. Die Zeiten werden in Einheiten zu 0.02 Sekunden gemessen.
`CLS`	Füllen des Bildschirmbereiches mit der Farbe des durch PAPER bestimmten Farbstiftes.
`CLG F`	Füllen des Grafikbildschirmes mit der dem Farbstift F zugeordneten Farbe.
`TAG`	Ermöglicht mit dem PRINT-Kommando auf Grafik-Cursor-Positionen Text und Symbole auszugeben.
`TAGOFF`	Beendet den TAG-Befehl.
`PLOT X,Y,F`	Setzt einen Punkt in der dem Farbstift F zugeordneten Farbe an die durch X,Y festgelegte Stelle.
`DRAW X,Y,F`	Zeichnet eine Linie von der aktuellen Cursor-Position zu der durch X,Y festgelegten Stelle in der dem Farbstift F zugeordneten Farbe.
`MOVE X,Y`	Positioniert den Grafik-Cursor auf die durch X,Y festgelegte Stelle.
`TEST (X,Y)`	Das TEST-Kommando liefert die Farbstiftnummer, die an der Stelle X,Y verwendet wurde.
`ORIGIN X,Y`	Setzt den Startpunkt des Grafik-Cursors an die Stelle X,Y. Dieser Punkt hat danach die Koordinaten (0,0).
`ORIGIN X0,Y0,L,R,O,U`	Legt einen Grafik-Bildschirmbereich (Fenster) fest. Dabei bestimmen L,R,O,U die Größe dieses Bereiches und X0,Y0

	die Koordinaten der linken unteren Ecke nach Ausführung des Kommandos.
`WINDOW #A,L,R,O,U`	Legt einen Text-Bildschirmteilbereich (Text-Fenster) als ein mit #A ansprechbares Ein-/Ausgabegerät fest. L, R. O, U beziehen sich dabei auf Zeilen und Spalten und hängen daher vom Bildschirmmodus ab.

Strukturierte Programmierung

`IF...THEN...ELSE...`	Falls (IF) die Bedingung wahr, dann (THEN) Anweisung ausführen, ansonsten (ELSE) Anweisung.
`WHILE....WEND`	Die Schleife von WHILE bis WEND wird so lange durchlaufen, wie die Bedingung hinter WHILE wahr ist.

Fehlerbehandlung

`ON ERROR GOTO`	Tritt ein Programmfehler auf, so springt das Programm in die gewünschte Zeile. Hier kann dann die Fehlerbehandlung mit Hilfe der Variablen ERR = Fehlernummer und ERL = Zeilennummer beginnen.
`RESUME` Zeilennummer `RESUME NEXT`	Das Programm macht in der angegebenen Zeilennummer oder in der nächsten Zeile weiter, falls es durch die Fehlerbehandlung ON ERROR GOTO unterbrochen wurde.
`ERROR` Fehlernummer	Führt die für den Fehler mit dieser Nummer vorgesehene Behandlung aus. Falls die Fehlernummer nicht von SCHNEIDER-BASIC verwendet wird, wird der Fehler nur angezeigt und das Programm abgebrochen.

Register

Einer sehr gut verständlichen Programmbeschreibung folgt jeweils das komplette Listing und die Abbildung der entsprechenden Grafik. Ohne tiefergehende Kenntnisse vorauszusetzen, werden elementare Programmierungstechniken wie z. B. die Bildschirmfenstergestaltung, das Schraffieren und das Zeichnen geometrischer Grundfiguren ausführlich erläutert und an vielen Abbildungen demonstriert. Optisch eindrucksvolle Demo-Grafiken in schönen Farbgebungen, eine Fülle von anwendungsorientierten Grafikprogrammen aus Mathematik und Physik, komfortable menügesteuerte Zeichenprogramme, 3D-Grafiken sowie Programme zur grafischen Lösung von Differentialgleichungen zeigen, daß alle wesentlichen Grafikmöglichkeiten vom Autor behandelt werden. Die Programme kommen mit dem Befehlssatz von SCHNEIDER BASIC aus und sind vom Leser leicht nach eigenen Wünschen zu ändern.

ISBN 3 - 14 - 1388 18 - 0

GRAFIK AUF DEM CPC 464

westermann COMPUTERBUCH

AMSTRAD CPC

OCR

MÉMOIRE ÉCRITE
MEMORY ENGRAVED
MEMORIA ESCRITA



https://acpc.me/